ESOF 423
Software Engineering Applications
Portfolio Report

Mason Dinardi, Nathan Rubino, Peyton Ellis

Section 1: Program

Controllers

```
'use
strict'

/** @typedef {import('@adonisjs/framework/src/Request')}
Request */
/** @typedef {import('@adonisjs/framework/src/Response')}
Response */
/** @typedef {import('@adonisjs/framework/src/View')} View */


/**
* Resourceful controller for interacting with seatingcharts
*/


const SeatingChart = use('App/Models/SeatingChart')
const { validate } = use('Validator')
var date


class SeatingChartController {


 async create ({ request, params }) {
 }


 async edit ({ params, view }) {
   const seating_chart = await SeatingChart.find(params.id)


   return view.render('edit-ticket', {
```

```
    seating_chart: seating_chart
  })
}

async update ({ params, request, response }) {

  const validation = await validate(request.all(), {
    Name: 'required',
    Phone_Number: 'required',
    Seat_type: 'required',
    Seats_rsv: 'required'
  })
  if(validation.fails()){
    return response.redirect('back')
  }

  const seating_chart = await SeatingChart.find(params.id)

  seating_chart.Name = request.all().Name
  seating_chart.Phone_Number = request.all().Phone_Number
  seating_chart.Seat_type = request.all().Seat_type
  seating_chart.Seats_rsv = request.all().Seats_rsv

  await seating_chart.save()
  return response.redirect('/future-shows')
}

async delete({ response, params}){
  const seating_chart = await SeatingChart.find(params.id)
```

```
    await seating_chart.delete()
    return response.redirect('back')
  }

}

module.exports = SeatingChartController
```

```
'use strict'

//import Database from '@ioc:Adonis/Lucid/Database'
const Show = use('App/Models/Show')
const { validate } = use('Validator')
var date
var showToPrint

class ShowController {

 /*
 * query to retrieve all data for the future-shows page
 */
 async index({view}){

   const future = await Show
     .query()
     .select('id', 'Show_title', 'Show_date')
```

```
      .from('shows')
      .where('isPast', 0)
      .fetch()


    return view.render('future-shows', {
      shows: future.toJSON()
    })
  }


  /*
  * retrieves all shows that have been marked as completed
  */
  async pastIndex({view}){
    const past = await Show
      .query()
      .select('id', 'Show_title', 'Show_date')
      .from('shows')
      .where('isPast', 1)
      .fetch()


    return view.render('previous-shows', {
      shows: past.toJSON()
    })
  }


  /*
  * gets all information for add-tickets page to display tickets for
selected show
  */
  async details({ params, view }){
```

```
    const show = await Show.find(params.id)


    date = show.Show_date


    const current_show = await Show
      .query()
      .select('*')
      .from('seating_charts')
      .where('Show', '=', show.Show_date)
      .fetch()


    return view.render('add-ticket', {
      show: show.toJSON(),
      seating_charts: current_show.toJSON()
    })
  }


  /*
  * Creates new show with validators.
  */
  async create ({ request, response, session }) {
    //validate input
    const validation = await validate(request.all(), {
      Show_title: 'required',
      Show_date: 'required'
    })
    if(validation.fails()){
      session.flash({ notification: 'Show failed to add' })
      return response.redirect('back')
```

```
  }



  const data = request.only(['Show_title', 'Show_date'])
  data.isPast = 0


  // save and get instance back
  const show = await Show.create(data)
  session.flash({ notification: 'Show Added!' })
  return response.redirect('back')
}


/*
* Helper function to declare a show completed
*/
async isPast ({params, response}){
  const show = await Show.find(params.id)


  show.isPast = 1


  await show.save()
  return response.redirect('back')
}


async oops ({params, response}){
  const show = await Show.find(params.id)


  show.isPast = 0
```

```
  await show.save()
  return response.redirect('back')
}

/*
* renders edit-show with auto-filled data
*/
async edit({ params, view }){
  const show = await Show.find(params.id)

  return view.render('edit-show', {
    show: show
  })
}

/*
* Actually updates show in database
*/
async update ({ response, request, params}){

  const validation = await validate(request.all(), {
    Show_title: 'required',
    Show_date: 'required'
  })
  if(validation.fails()){
    return response.redirect('back')
  }
```

```
    const show = await Show.find(params.id)


    show.Show_title = request.all().Show_title
    show.Show_date = request.all().Show_date


    await show.save()
    return response.redirect('/future-shows')
  }


  /*
  * yall can figure this one out im sure
  */
  async delete({ response, params}){
    const show = await Show.find(params.id)


    await show.delete()
    return response.redirect('back')
  }


  /*
  * inserts new ticket into seating_charts table. Only way I could
dynamically insert dates
  */
  async insert_ticket({request, response, session}) {
    const SeatingChart = use('App/Models/SeatingChart')


    const validation = await validate(request.all(), {
      Name: 'required',
      Phone_Number: 'required',
      Seat_type: 'required',
```

```
    Seats_rsv: 'required'
  })
  if(validation.fails()){
    return response.redirect('back')
  }


  const data = await request.only(['Name', 'Phone_Number', 'Seat_type',
'Seats_rsv'])
  data.Show = date


  const namesQuery = await Show
    .query()
    .select('Name', 'Seats_rsv')
    .from('seating_charts')
    .where('Show', date)
    /*.whereExists(function() {
    this.select('1').from('seating_charts').whereRaw('Show = ?',
date).whereRaw('Name = ?', data.Name)
  })*/
    .fetch()


  var names = namesQuery.toJSON()
  console.log(names)
  var flag = false
  for (var i = 0; i < names.length; ++i) {
    var name = names[i]
    if(name.Name == data.Name || name.Seats_rsv == data.Seats_rsv) {
      flag = true
    }
  }
```

```
    if(flag) {
      session.flash({ notification: 'Name has already been used OR seat has already been booked' })
      return response.redirect('back')
    }

    session.flash({ notification: 'Ticket Added!'})
    await SeatingChart.create(data)
    return response.redirect('back')
  }

  /*
  * displays all uncompleted shows on print-tickets page
  */
  async print_display( {view}) {

    const future = await Show
      .query()
      .select('id', 'Show_title', 'Show_date')
      .from('shows')
      .where('isPast', 0)
      .fetch()

    return view.render('print-tickets', {
      shows: future.toJSON()
    })
  }

  /*
  * displays all tickets to be printed for a given show
```

```
 */
 async print_tickets( {view, params}) {
   const show = await Show.find(params.id)


   date = show.Show_date
   showToPrint = await Show.find(params.id)


   console.log(showToPrint)


   const current_show = await Show
     .query()
     .select('*')
     .from('seating_charts')
     .where('Show', '=', show.Show_date)
     .fetch()


   return view.render('print', {
     show: show.toJSON(),
     seating_charts: current_show.toJSON()
   })
 }


 /*
* gets all information to be formatted for a given ticket
*/
 async ticket( {view, params}) {
   const SeatingChart = use('App/Models/SeatingChart')
   const ticket = await SeatingChart.find(params.id)
```

```
    const current_show = showToPrint
    console.log(current_show)


    return view.render('print-page', {
      seating_chart: ticket.toJSON(),
      show: current_show.toJSON()
    })
  }
}


module.exports = ShowController
```

Middleware

```
'use strict'


class ConvertEmptyStringsToNull {
 async handle ({ request }, next) {
   if (Object.keys(request.body).length) {
     request.body = Object.assign(
       ...Object.keys(request.body).map(key => ({
         [key]: request.body[key] !== '' ? request.body[key] :
null
       }))
     )
   }


   await next()
```

```
  }
}


module.exports = ConvertEmptyStringsToNull
```

Models

```
'use
strict'


class NoTimestamp {
 register (Model) {

Object.defineProperties(Model, {
     createdAtColumn: {
       get: () => null,
     },
     updatedAtColumn: {
       get: () => null,
     },
   })
 }
}


module.exports = NoTimestamp
```

```
'use strict'
```

```
/** @type {typeof
import('@adonisjs/lucid/src/Lucid/Model')} */
const Model = use('Model')


class SeatingChart extends Model {
}


module.exports =
```

```
'use
strict'
```

```
/** @type {typeof
import('@adonisjs/lucid/src/Lucid/Model')} */
const Model = use('Model')


class Show extends Model {


 seating_charts(){
   return this.hasMany('App/Models/SeatingChart')
 }


}




module.exports = Show
```

```
'use
strict'


/** @type {import('@adonisjs/framework/src/Hash')} */
const Hash = use('Hash')


/** @type {typeof import('@adonisjs/lucid/src/Lucid/Model')} */
const Model = use('Model')


class User extends Model {
 static boot () {
   super.boot()


   /**
    * A hook to hash the user password before saving
    * it to the database.
    */
   this.addHook('beforeSave', async (userInstance) => {
     if (userInstance.dirty.password) {
       userInstance.password = await
Hash.make(userInstance.password)
     }
   })
 }


 /**
  * A relationship on tokens is required for auth to
  * work. Since features like `refreshTokens` or
  * `rememberToken` will be saved inside the
```

```
   * tokens table.
   *
   * @method tokens
   *
   * @return {Object}
   */
  tokens () {
    return this.hasMany('App/Models/Token')
  }
}


module.exports = User
```

Config

```
'use strict'


/** @type {import('@adonisjs/framework/src/Env')} */
const Env = use('Env')


module.exports = {


 /*

|--------------------------------------------------------------------------
-----
 | Application Name

|--------------------------------------------------------------------------
```

```
-----
 |
 | This value is the name of your application and can be used when you
 | need to place the application's name in a email, view or
 | other location.
 |
 */

 name: Env.get('APP_NAME', 'AdonisJs'),

 /*
|--------------------------------------------------------------------------
-----
 | App Key
|--------------------------------------------------------------------------
-----
 |
 | App key is a randomly generated 16 or 32 characters long string
required
 | to encrypted cookies, sessions and other sensitive data.
 |
 */
 appKey: Env.getOrFail('APP_KEY'),

 http: {
   /*
|--------------------------------------------------------------------------
-----
   | Allow Method Spoofing
|--------------------------------------------------------------------------
-----
```

```
   |
   | Method spoofing allows you to make requests by spoofing the http
verb.
   | Which means you can make a GET request but instruct the server to
   | treat as a POST or PUT request. If you want this feature, set the
   | below value to true.
   |
   */
   allowMethodSpoofing: true,


   /*
|--------------------------------------------------------------------------
-----
   | Trust Proxy
|--------------------------------------------------------------------------
-----
   |
   | Trust proxy defines whether X-Forwarded-* headers should be
trusted or not.
   | When your application is behind a proxy server like nginx, these
values
   | are set automatically and should be trusted. Apart from setting
it
   | to true or false Adonis supports a handful of ways to allow proxy
   | values. Read documentation for that.
   |
   */
   trustProxy: false,


   /*
|--------------------------------------------------------------------------
-----
   | Subdomains
```

```
|--------------------------------------------------------------------------
-----
   |
   | Offset to be used for returning subdomains for a given request.
For
   | majority of applications it will be 2, until you have nested
   | sudomains.
   | cheatsheet.adonisjs.com      - offset - 2
   | virk.cheatsheet.adonisjs.com - offset - 3
   |
   */
   subdomainOffset: 2,


   /*

|--------------------------------------------------------------------------
-----
   | JSONP Callback

|--------------------------------------------------------------------------
-----
   |
   | Default jsonp callback to be used when callback query string is
missing
   | in request url.
   |
   */
   jsonpCallback: 'callback',




   /*

|--------------------------------------------------------------------------
-----
```

```
    | Etag
    |
    |--------------------------------------------------------------------------
    |
    | Set etag on all HTTP responses. In order to disable for selected routes,
    | you can call the `response.send` with an options object as follows.
    |
    | response.send('Hello', { ignoreEtag: true })
    |
    */
    etag: false
  },


  views: {
    /*
    |--------------------------------------------------------------------------
    | Cache Views
    |--------------------------------------------------------------------------
    |
    | Define whether or not to cache the compiled view. Set it to true in
    | production to optimize view loading time.
    |
    */
    cache: Env.get('CACHE_VIEWS', true)
  },


  static: {
```

```
   /*

  |------------------------------------------------------------------------
  -----
   | Dot Files

  |------------------------------------------------------------------------
  -----
   |
   | Define how to treat dot files when trying to serve static
  resources.
   | By default it is set to ignore, which will pretend that dotfiles
   | do not exist.
   |
   | Can be one of the following
   | ignore, deny, allow
   |
   */
   dotfiles: 'ignore',


   /*

  |------------------------------------------------------------------------
  -----
   | ETag

  |------------------------------------------------------------------------
  -----
   |
   | Enable or disable etag generation
   |
   */
   etag: true,


   /*
```

```
|--------------------------------------------------------------------------
-----
   | Extensions

|--------------------------------------------------------------------------
-----
   |
   | Set file extension fallbacks. When set, if a file is not found,
the given
   | extensions will be added to the file name and search for. The
first
   | that exists will be served. Example: ['html', 'htm'].
   |
   */
   extensions: false
 },


 locales: {
   /*

|--------------------------------------------------------------------------
-----
   | Loader

|--------------------------------------------------------------------------
-----
   |
   | The loader to be used for fetching and updating locales. Below is
the
   | list of available options.
   |
   | file, database
   |
   */
   loader: 'file',
```

```
   /*
  |--------------------------------------------------------------------------
  -----
   | Default Locale
  |--------------------------------------------------------------------------
  -----
   |
   | Default locale to be used by Antl provider. You can always switch
  drivers
   | in runtime or use the official Antl middleware to detect the
  driver
   | based on HTTP headers/query string.
   |
   */
   locale: 'en'
 },


 logger: {
   /*
  |--------------------------------------------------------------------------
  -----
   | Transport
  |--------------------------------------------------------------------------
  -----
   |
   | Transport to be used for logging messages. You can have multiple
   | transports using same driver.
   |
   | Available drivers are: `file` and `console`.
   |
```

```
    */
    transport: 'console',



    /*

|--------------------------------------------------------------------------
-----
    | Console Transport

|--------------------------------------------------------------------------
-----
    |
    | Using `console` driver for logging. This driver writes to
`stdout`
    | and `stderr`
    |
    */
    console: {
      driver: 'console',
      name: 'adonis-app',
      level: 'info'
    },



    /*

|--------------------------------------------------------------------------
-----
    | File Transport

|--------------------------------------------------------------------------
-----
    |
    | File transport uses file driver and writes log messages for a
given
    | file inside `tmp` directory for your app.
```

```
    |
    | For a different directory, set an absolute path for the filename.
    |
    */
    file: {
      driver: 'file',
      name: 'adonis-app',
      filename: 'adonis.log',
      level: 'info'
    }
  },


  /*
  |--------------------------------------------------------------------------
  | Generic Cookie Options
  |--------------------------------------------------------------------------
  |
  | The following cookie options are generic settings used by AdonisJs to create
  | cookies. However, some parts of the application like `sessions` can have
  | seperate settings for cookies inside `config/session.js`.
  |
  */
  cookie: {
    httpOnly: true,
    sameSite: false,
    path: '/',
    maxAge: 7200
  }
```

```
}
```

```
'use
strict'


/** @type {import('@adonisjs/framework/src/Env')} */
const Env = use('Env')


module.exports = {
 /*

|--------------------------------------------------------------------------
------
 | Authenticator

|--------------------------------------------------------------------------
------
 |
 | Authentication is a combination of serializer and scheme with
extra
 | config to define on how to authenticate a user.
 |
 | Available Schemes - basic, session, jwt, api
 | Available Serializers - lucid, database
 |
 */
 authenticator: 'session',


 /*

|--------------------------------------------------------------------------
```

```
            ------
             | Session

            |-----------------------------------------------------------------
            ------
             |
             | Session authenticator makes use of sessions to authenticate a
            user.
             | Session authentication is always persistent.
             |
             */
             session: {
               serializer: 'lucid',
               model: 'App/Models/User',
               scheme: 'session',
               uid: 'email',
               password: 'password'
             },


             /*

            |-----------------------------------------------------------------
            ------
             | Basic Auth

            |-----------------------------------------------------------------
            ------
             |
             | The basic auth authenticator uses basic auth header to
            authenticate a
             | user.
             |
             | NOTE:
             | This scheme is not persistent and users are supposed to pass
             | login credentials on each request.
```

```
    |
    */
    basic: {
      serializer: 'lucid',
      model: 'App/Models/User',
      scheme: 'basic',
      uid: 'email',
      password: 'password'
    },


    /*
    |--------------------------------------------------------------------------
    | Jwt
    |--------------------------------------------------------------------------
    |
    | The jwt authenticator works by passing a jwt token on each HTTP request
    | via HTTP `Authorization` header.
    |
    */
    jwt: {
      serializer: 'lucid',
      model: 'App/Models/User',
      scheme: 'jwt',
      uid: 'email',
      password: 'password',
      options: {
        secret: Env.get('APP_KEY')
      }
```

```
  },


  /*

 |--------------------------------------------------------------------------
 ------
  | Api

 |--------------------------------------------------------------------------
 ------
  |
  | The Api scheme makes use of API personal tokens to authenticate a
 user.
  |
  */
  api: {
    serializer: 'lucid',
    model: 'App/Models/User',
    scheme: 'api',
    uid: 'email',
    password: 'password'
  }
 }
```

```
'use
strict'


module.exports = {
  /*

 |--------------------------------------------------------------------------
 -----
```

```
 | JSON Parser

|-----------------------------------------------------------------------
-----
 |
 | Below settings are applied when request body contains JSON payload.
If
 | you want body parser to ignore JSON payload, then simply set
`types`
 | to an empty array.
 */
 json: {
   /*

|-----------------------------------------------------------------------
-----
   | limit

|-----------------------------------------------------------------------
-----
   |
   | Defines the limit of JSON that can be sent by the client. If
payload
   | is over 1mb it will not be processed.
   |
   */
   limit: '1mb',



   /*

|-----------------------------------------------------------------------
-----
   | strict

|-----------------------------------------------------------------------
-----
   |
```

```
    | When `scrict` is set to true, body parser will only parse Arrays
and
    | Object. Otherwise everything parseable by `JSON.parse` is parsed.
    |
    */
    strict: true,


    /*
|--------------------------------------------------------------------------
-----
    | types
|--------------------------------------------------------------------------
-----
    |
    | Which content types are processed as JSON payloads. You are free
to
    | add your own types here, but the request body should be parseable
    | by `JSON.parse` method.
    |
    */
    types: [
      'application/json',
      'application/json-patch+json',
      'application/vnd.api+json',
      'application/csp-report'
    ]
  },


  /*
|--------------------------------------------------------------------------
-----
```

```
 | Raw Parser

|-----------------------------------------------------------------------
-----
 |
 |
 |
 */
 raw: {
   types: [
     'text/*'
   ]
 },


 /*

|-----------------------------------------------------------------------
-----
 | Form Parser

|-----------------------------------------------------------------------
-----
 |
 |
 |
 */
 form: {
   types: [
     'application/x-www-form-urlencoded'
   ]
 },


 /*
```

```
|--------------------------------------------------------------------------
-----
 | Files Parser

|--------------------------------------------------------------------------
-----
 |
 |
 |
 */
 files: {
   types: [
     'multipart/form-data'
   ],


   /*

|--------------------------------------------------------------------------
-----
   | Max Size

|--------------------------------------------------------------------------
-----
   |
   | Below value is the max size of all the files uploaded to the
server. It
   | is validated even before files have been processed and hard
exception
   | is thrown.
   |
   | Consider setting a reasonable value here, otherwise people may
upload GB's
   | of files which will keep your server busy.
   |
   | Also this value is considered when `autoProcess` is set to true.
```

```
    |
    */
    maxSize: '20mb',


    /*
|--------------------------------------------------------------------------
-----
    | Auto Process
|--------------------------------------------------------------------------
-----
    |
    | Whether or not to auto-process files. Since HTTP servers handle
files via
    | couple of specific endpoints. It is better to set this value off
and
    | manually process the files when required.
    |
    | This value can contain a boolean or an array of route patterns
    | to be autoprocessed.
    */
    autoProcess: true,


    /*
|--------------------------------------------------------------------------
-----
    | Process Manually
|--------------------------------------------------------------------------
-----
    |
    | The list of routes that should not process files and instead rely
on
    | manual process. This list should only contain routes when
```

```
autoProcess
   | is to true. Otherwise everything is processed manually.
   |
   */
   processManually: []



   /*

|--------------------------------------------------------------------------
-----
   | Temporary file name

|--------------------------------------------------------------------------
-----
   |
   | Define a function, which should return a string to be used as the
   | tmp file name.
   |
   | If not defined, Bodyparser will use `uuid` as the tmp file name.
   |
   | To be defined as. If you are defining the function, then do make
sure
   | to return a value from it.
   |
   | tmpFileName () {
   |   return 'some-unique-value'
   | }
   |
   */
 }
}
```

```
'use
strict'

            module.exports = {
             /*
            |--------------------------------------------------------------------------
            -----
             | Origin
            |--------------------------------------------------------------------------
            -----
             |
             | Set a list of origins to be allowed. The value can be one of the
            following
             |
             | Boolean: true - Allow current request origin
             | Boolean: false - Disallow all
             | String - Comma seperated list of allowed origins
             | Array - An array of allowed origins
             | String: * - A wildcard to allow current request origin
             | Function - Receives the current origin and should return one of the
            above values.
             |
             */
             origin: false,

             /*
            |--------------------------------------------------------------------------
            -----
             | Methods
            |--------------------------------------------------------------------------
            -----
             |
```

```
 | HTTP methods to be allowed. The value can be one of the following
 |
 | String - Comma seperated list of allowed methods
 | Array - An array of allowed methods
 |
 */
 methods: ['GET', 'PUT', 'PATCH', 'POST', 'DELETE'],


 /*

|-------------------------------------------------------------------------
-----
 | Headers

|-------------------------------------------------------------------------
-----
 |
 | List of headers to be allowed via Access-Control-Request-Headers
header.
 | The value can be on of the following.
 |
 | Boolean: true - Allow current request headers
 | Boolean: false - Disallow all
 | String - Comma seperated list of allowed headers
 | Array - An array of allowed headers
 | String: * - A wildcard to allow current request headers
 | Function - Receives the current header and should return one of the
above values.
 |
 */
 headers: true,


 /*
```

```
|-----------------------------------------------------------------------
-----
 | Expose Headers

|-----------------------------------------------------------------------
-----
 |
 | A list of headers to be exposed via `Access-Control-Expose-Headers`
 | header. The value can be on of the following.
 |
 | Boolean: false - Disallow all
 | String: Comma seperated list of allowed headers
 | Array - An array of allowed headers
 |
 */
 exposeHeaders: false,


 /*

|-----------------------------------------------------------------------
-----
 | Credentials

|-----------------------------------------------------------------------
-----
 |
 | Define Access-Control-Allow-Credentials header. It should always be
a
 | boolean.
 |
 */
 credentials: false,


 /*
```

```
|--------------------------------------------------------------------------
-----
 | MaxAge

|--------------------------------------------------------------------------
-----
 |
 | Define Access-Control-Allow-Max-Age
 |
 */
 maxAge: 90
}
```

```
'use
strict'


/** @type {import('@adonisjs/framework/src/Env')} */
const Env = use('Env')


/** @type {import('@adonisjs/ignitor/src/Helpers')} */
const Helpers = use('Helpers')


module.exports = {
 /*

|--------------------------------------------------------------------------
----
 | Default Connection

|--------------------------------------------------------------------------
----
```

```
  |
  | Connection defines the default connection settings to be used while
  | interacting with SQL databases.
  |
  */
  connection: Env.get('DB_CONNECTION', 'sqlite'),


  /*
 |--------------------------------------------------------------------------
 | Sqlite
 |--------------------------------------------------------------------------
 |
 | Sqlite is a flat file database and can be good choice under development
 | environment.
 |
 | npm i --save sqlite3
 |
 */
 sqlite: {
   client: 'sqlite3',
   connection: {
     filename: Helpers.databasePath(`${Env.get('DB_DATABASE', 'development')}.sqlite`)
   },
   useNullAsDefault: true
 },


 /*
```

```
|--------------------------------------------------------------------------
----
 | MySQL
 |
|--------------------------------------------------------------------------
----
 |
 | Here we define connection settings for MySQL database.
 |
 | npm i --save mysql
 |
 */
 mysql: {
   client: 'mysql',
   connection: {
     host: Env.get('DB_HOST', 'localhost'),
     port: Env.get('DB_PORT', ''),
     user: Env.get('DB_USER', 'root'),
     password: Env.get('DB_PASSWORD', ''),
     database: Env.get('DB_DATABASE', 'adonis')
   }
 },


 /*
|--------------------------------------------------------------------------
----
 | PostgreSQL
 |
|--------------------------------------------------------------------------
----
 |
 | Here we define connection settings for PostgreSQL database.
 |
```

```
| npm i --save pg
|
*/
pg: {
  client: 'pg',
  connection: {
    host: Env.get('DB_HOST', 'localhost'),
    port: Env.get('DB_PORT', ''),
    user: Env.get('DB_USER', 'root'),
    password: Env.get('DB_PASSWORD', ''),
    database: Env.get('DB_DATABASE', 'adonis')
  }
}
}
```

```
'use
strict'

/** @type {import('@adonisjs/framework/src/Env')} */
const Env = use('Env')


module.exports = {
 /*
|--------------------------------------------------------------------------
-----
 | Driver
|--------------------------------------------------------------------------
-----
```

```
 |
 | Driver to be used for hashing values. The same driver is used by
the
 | auth module too.
 |
 */
 driver: Env.get('HASH_DRIVER', 'bcrypt'),


 /*

|--------------------------------------------------------------------------
-----
 | Bcrypt

|--------------------------------------------------------------------------
-----
 |
 | Config related to bcrypt hashing.
https://www.npmjs.com/package/bcrypt
 | package is used internally.
 |
 */
 bcrypt: {
   rounds: 10
 },


 /*

|--------------------------------------------------------------------------
-----
 | Argon

|--------------------------------------------------------------------------
-----
 |
```

```
 | Config related to argon. https://www.npmjs.com/package/argon2
package is
 | used internally.
 |
 | Since argon is optional, you will have to install the dependency
yourself
 |

|=============================================================================
=======
 | npm i argon2

|=============================================================================
=======
 |
 */
 argon: {
   type: 1
 }
}
```

```
'use
strict'


/** @type {import('@adonisjs/framework/src/Env')} */
const Env = use('Env')


module.exports = {
 /*

|-----------------------------------------------------------------------
------
```

```
 | Session Driver
 |
 |--------------------------------------------------------------------------
 |
 | The session driver to be used for storing session values. It can be
 | cookie, file or redis.
 |
 | For `redis` driver, make sure to install and register `@adonisjs/redis`
 |
 */
 driver: Env.get('SESSION_DRIVER', 'cookie'),


 /*
 |--------------------------------------------------------------------------
 | Cookie Name
 |--------------------------------------------------------------------------
 |
 | The name of the cookie to be used for saving session id. Session ids
 | are signed and encrypted.
 |
 */
 cookieName: 'adonis-session',


 /*
 |--------------------------------------------------------------------------
 | Clear session when browser closes
```

```
|-------------------------------------------------------------------
------
 |
 | If this value is true, the session cookie will be temporary and
will be
 | removed when browser closes.
 |
 */
 clearWithBrowser: true,


 /*

|-------------------------------------------------------------------
------
 | Session age

|-------------------------------------------------------------------
------
 |
 | This value is only used when `clearWithBrowser` is set to false.
The
 | age must be a valid https://npmjs.org/package/ms string or should
 | be in milliseconds.
 |
 | Valid values are:
 |  '2h', '10d', '5y', '2.5 hrs'
 |
 */
 age: '2h',


 /*

|-------------------------------------------------------------------
------
```

```
 | Cookie options

|--------------------------------------------------------------------------
------
 |
 | Cookie options defines the options to be used for setting up
session
 | cookie
 |
 */
 cookie: {
   httpOnly: true,
   sameSite: false,
   path: '/'
 },


 /*

|--------------------------------------------------------------------------
------
 | Sessions location

|--------------------------------------------------------------------------
------
 |
 | If driver is set to file, we need to define the relative location
from
 | the temporary path or absolute url to any location.
 |
 */
 file: {
   location: 'sessions'
 },
```

```
  /*
  |--------------------------------------------------------------------------
  | Redis config
  |--------------------------------------------------------------------------
  |
  | The configuration for the redis driver. By default we reference it from
  | the redis file. But you are free to define an object here too.
  |
  */
  redis: {
    host: '127.0.0.1',
    port: 6379,
    password: null,
    db: 0,
    keyPrefix: ''
  }
}
```

```
'use strict'

module.exports = {
  /*
  |--------------------------------------------------------------------------
  | Content Security Policy
```

```
|-------------------------------------------------------------------------
----
 |
 | Content security policy filters out the origins not allowed to
execute
 | and load resources like scripts, styles and fonts. There are wide
 | variety of options to choose from.
 */
 csp: {
   /*

|-------------------------------------------------------------------------
----
   | Directives

|-------------------------------------------------------------------------
----
   |
   | All directives are defined in camelCase and here is the list of
   | available directives and their possible values.
   |
   | https://content-security-policy.com
   |
   | @example
   | directives: {
   |   defaultSrc: ['self', '@nonce', 'cdnjs.cloudflare.com']
   | }
   |
   */
   directives: {
   },
   /*

|-------------------------------------------------------------------------
----
```

```
   | Report only

|--------------------------------------------------------------------
----
   |
   | Setting `reportOnly=true` will not block the scripts from running
and
   | instead report them to a URL.
   |
   */
   reportOnly: false,
   /*

|--------------------------------------------------------------------
----
   | Set all headers

|--------------------------------------------------------------------
----
   |
   | Headers staring with `X` have been depreciated, since all major
browsers
   | supports the standard CSP header. So its better to disable
deperciated
   | headers, unless you want them to be set.
   |
   */
   setAllHeaders: false,



   /*

|--------------------------------------------------------------------
----
   | Disable on android

|--------------------------------------------------------------------
----
```

```
    |
    | Certain versions of android are buggy with CSP policy. So you can
set
    | this value to true, to disable it for Android versions with buggy
    | behavior.
    |
    | Here is an issue reported on a different package, but helpful to
read
    | if you want to know the behavior.
https://github.com/helmetjs/helmet/pull/82
    |
    */
    disableAndroid: true
  },


  /*

|-----------------------------------------------------------------------
----
  | X-XSS-Protection

|-----------------------------------------------------------------------
----
  |
  | X-XSS Protection saves applications from XSS attacks. It is adopted
  | by IE and later followed by some other browsers.
  |
  | Learn more at
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protect
ion
  |
  */
  xss: {
    enabled: true,
    enableOnOldIE: false
```

```
 },


 /*

|--------------------------------------------------------------------------
----
 | Iframe Options

|--------------------------------------------------------------------------
----
 |
 | xframe defines whether or not your website can be embedded inside an
 | iframe. Choose from one of the following options.
 | @available options
 | DENY, SAMEORIGIN, ALLOW-FROM http://example.com
 |
 | Learn more at
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Optio
ns
 */
 xframe: 'DENY',


 /*

|--------------------------------------------------------------------------
----
 | No Sniff

|--------------------------------------------------------------------------
----
 |
 | Browsers have a habit of sniffing content-type of a response. Which
means
 | files with .txt extension containing Javascript code will be
executed as
 | Javascript. You can disable this behavior by setting nosniff to
false.
```

```
 |
 | Learn more at
https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Optio
ns
 |
 */
 nosniff: true,


 /*

|-----------------------------------------------------------------------
----
 | No Open

|-----------------------------------------------------------------------
----
 |
 | IE users can execute webpages in the context of your website, which
is
 | a serious security risk. Below option will manage this for you.
 |
 */
 noopen: true,


 /*

|-----------------------------------------------------------------------
----
 | CSRF Protection

|-----------------------------------------------------------------------
----
 |
 | CSRF Protection adds another layer of security by making sure,
actionable
 | routes does have a valid token to execute an action.
```

```
  |
  */
  csrf: {
    enable: true,
    methods: ['POST', 'PUT', 'DELETE'],
    filterUris: [],
    cookieOptions: {
      httpOnly: false,
      sameSite: true,
      path: '/',
      maxAge: 7200
    }
  }
}
```

Migrations

```
'use
strict'


 /** @type {import('@adonisjs/lucid/src/Schema')} */
 const Schema = use('Schema')


 class UserSchema extends Schema {
  up () {
    this.create('users', (table) => {
      table.increments()
      table.string('username',
 80).notNullable().unique()
      table.string('email',
```

```
254).notNullable().unique()
      table.string('password', 60).notNullable()
      table.timestamps()
    })
  }

  down () {
    this.drop('users')
  }
}

module.exports = UserSchema
```

```
'use strict'

/** @type {import('@adonisjs/lucid/src/Schema')} */
const Schema = use('Schema')

class TokensSchema extends Schema {
  up () {
    this.create('tokens', (table) => {
      table.increments()
table.integer('user_id').unsigned().references('id').inTable('users')
      table.string('token', 255).notNullable().unique().index()
      table.string('type', 80).notNullable()
      table.boolean('is_revoked').defaultTo(false)
```

```
      table.timestamps()
    })
  }

  down () {
    this.drop('tokens')
  }
}

module.exports = TokensSchema
```

```
'use strict'

/** @type {import('@adonisjs/lucid/src/Schema')} */
const Schema = use('Schema')

class ShowsSchema extends Schema {
  up () {
    this.create('shows', (table) => {
      table.increments('id')
      table.string('Show_title')
      table.string('Show_date').unique()
      table.boolean('isPast')
      table.timestamps()
    })
  }
```

```
    down () {
      this.drop('shows')
    }
  }

  module.exports = ShowsSchema
```

```
'use strict'

/** @type {import('@adonisjs/lucid/src/Schema')} */
const Schema = use('Schema')

class SeatingChartSchema extends Schema {
 up () {
   this.create('seating_charts', (table) => {
     table.increments('id')
     table.string('Name')
     table.string('Phone_Number')
     table.string('Seat_type')
     table.string('Seats_rsv')

table.string('Show').references('Show_date').inTable('shows').onUpdate('CASCADE').onDelete('CASCADE')
     table.timestamps()
   })
 }
```

```
 down () {
   this.drop('seating_charts')
 }
}


module.exports = SeatingChartSchema
```

Stylesheets

```
@media print
{
    @page {
        /* The size is currently set to 'A4'; this probably
  needs to
        be changed in the future*/
        size: A4;
        margin: 0mm;
    }
    .no-print,
    .no-print * {
        display: none !important;
    }
}


a {
    color: #C87273;
}
```

```
body {
   text-align: center;
   background-color: #323639;
   color: #D6D6D7;
}

.no-print {
   padding: 30px;
}

.print-container {
   width: 66%;
   margin: auto;
   padding: 30px;
}

.ticket {
   padding-bottom: 15px;
   border: #C87273;
   border-style: dashed;
}

.ticket-element {
   padding: 5px;
}
```

```
/*
http://meyerweb.com/eric/tools/css/res
```

```
et/
  v2.0 | 20110126
  License: none (public domain)
*/

html, body, div, span, applet, object,
iframe,
h1, h2, h3, h4, h5, h6, p, blockquote,
pre,
a, abbr, acronym, address, big, cite,
code,
del, dfn, em, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr,
th, td,
article, aside, canvas, details, embed,
figure, figcaption, footer, header,
hgroup,
menu, nav, output, ruby, section,
summary,
time, mark, audio, video {
      margin: 0;
      padding: 0;
      border: 0;
      font-size: 100%;
      font: inherit;
      vertical-align: baseline;
}
/* HTML5 display-role reset for older
browsers */
article, aside, details, figcaption,
figure,
footer, header, hgroup, menu, nav,
section {
```

```
                display: block;
        }
        body {
                line-height: 1;
        }
        ol, ul {
                list-style: none;
        }
        blockquote, q {
                quotes: none;
        }
        blockquote:before, blockquote:after,
        q:before, q:after {
                content: '';
                content: none;
        }
        table {
                border-collapse: collapse;
                border-spacing: 0;
        }
```

```
a {
    color: black;
    text-decoration: none;
}

body {
    font-family: 'Verdana';
```

```
        background-color: #1A1A1D;
        padding: 10px;
        height: 100%;
    }


    button {
        margin: auto;
        background-color: #D6D6D7;
        border: none;
        padding: 10px 20px;
        text-align: center;
        text-decoration: none;
        font-size: 16px;
        cursor: pointer;
        padding: 5px;
        border-radius: 15px;
    }


    button:hover {
        background-color: grey;
    }


    form {
        margin-bottom: 0px;
    }


    html {
        height: 100%;
    }
```

```
h1 {
   color: #D6D6D7;
   text-align: center;
   font-size: large;
   padding: 15px;
   cursor: default;
}


input {
   float: right;
}


p {
   margin: auto;
   cursor: default;
}


.button-panel {
   padding: 45px;
   margin: auto;
   width: 33%;
}


.clear {
   clear: both;
}


.close {
   position: absolute;
   top: 15px;
```

```
        right: 35px;
        color: #f1f1f1;
        font-size: 40px;
        font-weight: bold;
        transition: 0.3s;
    }

    .close a {
        display: block;
    }

    .close:hover,
    .close:focus {
        color: #bbb;
        text-decoration: none;
        cursor: pointer;
    }

    .color1 {
        background-color: #950740;
    }

    .color2 {
        background-color: #C3073F;
    }

    .color3 {
        background-color: #6F2232;
    }
```

```
.container {
   width: 66%;
   margin: auto;
   padding: 15px;
   box-sizing: border-box;
   background-color: #4E4E50;
   border-radius: 25px 25px 0px 0px;
}


.custom_button {
   padding: 10px 20px;
   text-align: center;
   text-decoration: none;
   text-align: center;
   text-decoration: none;
}


.custom_button a {
   display: block;
}


.custom_button:hover {
   background-color: grey;
   cursor: pointer;
}


.database_entry {
   margin: 25px;
   padding: 15px;
```

```
}


.footer {
   background-color: #7E685A;
   width: 66%;
   margin: auto;
   padding-top: 5px;
   padding-bottom: 5px;
   border-radius: 0px 0px 25px 25px;
}


.footer_text {
   text-align: center;
   padding-top: 17px;
   padding-bottom: 3px;
}


.footer_text a {
   color: #950740;
}


.form_container {
   padding: 15px;
}


.future_shows {
   width: 75%;
   margin: auto;
   padding: 10px;
   color: #D6D6D7;
```

```
}


.left {
   float: left;
   width: 50%;
   height: 100%;
   box-sizing: border-box;
}


.left_color {
   background-color: #7E685A;
}


.left_container {
   padding: 10px;
}


.left_container a:hover {
   color: #C3073F;
}


.links {
   padding: 10px;
   margin: auto;
   width: 93%;
}


.links:hover {
   cursor: pointer;
```

```
}


.links a {
   display: block;
}


.logo {
   background: url("/vcplogo.jpg") no-repeat;
   width: 370px;
   height: 295px;
   margin: auto;
   margin-bottom: 15px;
   margin-top: 10px;
   text-align: center;
   opacity: 0;
   animation: slideUp 1s cubic-bezier(0.19, 1, 0.30, 1) 1.3s
forwards;
}


.modal {
   display: none;
   position: fixed;
   z-index: 1;
   padding-top: 100px;
   left: 0;
   top: 0;
   width: 100%;
   height: 100%;
   overflow: auto;
   background-color: rgb(0, 0, 0);
   background-color: rgba(0, 0, 0, 0.9);
```

```
}


.modal:hover {
   cursor: default;
}


.modal-content {
   margin: auto;
   display: block;
   width: 80%;
   max-width: 700px;
}


.modal-content,
#caption {
   -webkit-animation-name: zoom;
   -webkit-animation-duration: 0.6s;
   animation-name: zoom;
   animation-duration: 0.6s;
}


.myBtn_container {
   height: 36px;
   text-align: center;
   background-color: #C3073F;
}


.myBtn_container:hover {
   cursor: pointer;
}
```

```
.navbar {
   position: relative;
   text-align: center;
   padding: 10px;
}

.navbar a {
   padding: 10px;
}

.navbar a:hover {
   background-color: darkgrey;
   border-radius: 15px;
}

.navbar li {
   display: inline;
   background-color: #D6D6D7;
   border-radius: 15px;
}

.picture {
   margin: auto;
   width: 50%;
   display: block;
}

.previous_show_data {
```

```
        width: auto;
        margin: auto;
        cursor: default;
    }


    .print_section {
        width: 66%;
        margin: auto;
        height: 150px;
    }


    .right {
        float: right;
        width: 50%;
        box-sizing: border-box;
    }


    .right button {
        float: right;
    }


    .show_display {
        width: 66%;
        text-align: center;
        color: #D6D6D7;
        padding: 15px;
        margin: auto;
    }


    .show_display a {
```

```
        color: #D6D6D7;
    }


    .show_input {
        height: 30px;
        width: 66%;
        margin: auto;
        padding: 10px;
        font-size: larger;
    }


    .subtitle {
        padding: 15px;
        color: #D6D6D7;
        font-size: 17px;
        text-align: center;
        line-height: 2;
        opacity: 0;
        animation: slideUp 1s cubic-bezier(0.19, 1, 0.30, 1) 0.5s
    forwards;
    }


    .undo {
        width: 210px;
        height: 30px;
        margin: auto;
        padding: 5px;
    }


    .vertical_space {
```

```
        height: 70px;
    }


    #add_ticket_db {
        padding-bottom: 0px;
        margin-bottom: 0px;
        cursor: default;
    }


    #caption {
        margin: auto;
        display: block;
        width: 80%;
        max-width: 700px;
        text-align: center;
        color: #ccc;
        padding: 10px 0;
        height: 150px;
    }


    #myBtn {
        color: #D6D6D7;
        display: block;
    }


    #myBtn:hover {
        cursor: pointer;
    }


    #myImg {
```

```
    border-radius: 5px;
    cursor: pointer;
    transition: 0.3s;
}

#myImg:hover {
    opacity: 0.7;
}

#print_tickets_db {
    background-color: #7E685A;
}

#print_tickets_db a:hover {
    color: #950740;
}

#print_ticket_bg_color {
    background-color: #7E685A;
    margin-bottom: 0px;
    padding-bottom: 0px;
}

#submit_container {
    height: 28px;
    width: 90px;
    margin: auto;
    padding: 10px;
}
```

```
#submit_container button {
   width: 90px;
}


@keyframes slideUp {
   0% {
       transform: translateY(40px);
       opacity: 0;
   }
   50% {
       opacity: 0.2%;
   }
   100% {
       opacity: 1;
       transform: none;
   }
}


@keyframes zoom {
   from {
       transform: scale(0)
   }
   to {
       transform: scale(1)
   }
}


@-webkit-keyframes zoom {
   from {
       -webkit-transform: scale(0)
```

```
            }
            to {
                -webkit-transform: scale(1)
            }
        }
```

Views

```
<!DOCTYPE
html>
                <html lang="en">


                <head>
                    <meta charset="UTF-8" />
                    <meta name="viewport" content="width=device-width,
                initial-scale=1.0">
                    <meta http-equiv="X-UA-Compatible" content="ie=edge">
                    {{ style('reset') }}
                    {{ style('style') }}


                    @!section('extracss')


                    <title>
                        @!section('title')
                    </title>
                </head>


                <body>
                    <div class="container">
```

```
            <h1>
                @!section('title')
            </h1>


            @!section('content')
        </div>


        <div class="footer">
            <nav class="navbar">
                <ul>
                    <li><a href="/">Home</a></li>
                    <li><a href="/previous-shows">Previous
    Shows</a></li>
                    <li><a href="/future-shows">Future Shows</a></li>
                    <li><a href="/add-show">Add Shows</a></li>
                    <li><a href="/print-tickets">Print Tickets</a></li>
                </ul>
            </nav>
            <div class="footer_text">
                <br><a
    href="https://github.com/PeytonEllis/ESOF">GitHub</a>
            </div>
        </div>
    </body>


    </html>
```

```
@layout('layo
uts.main')
```

```
@section('extracss')

@endsection



@section('title')

   Add Shows

   @if(old('notification'))

       <div class="alert alert-success">

           {{ old('notification') }}

       </div>

   @endif

@endsection



@section('content')

   <head>

       <script
src="https://cdn.jsdelivr.net/npm/pikaday/pikaday.js"></script>
       <link rel="stylesheet" type="text/css"
href="https://cdn.jsdelivr.net/npm/pikaday/css/pikaday.css">
   </head>

   <body>

       <div class="future_shows color1">

           <form action="{{ route('ShowController.create') }}"
method="POST" autocomplete="off">
               {{ csrfField() }}



               <div id="show_input_container" class="color3">

                   <div class="show_input">

                       <label for="Show_title">Show Title:</label>

                       <input type="text" name="Show_title">

                   </div>

                   <div class="show_input">
```

```
                                <label for="Show_date">Show Date:</label>
                                <input type="text" id="datepicker"
name="Show_date">
                            </div>


                            <div id="submit_container">
                                <button type="submit">Submit</button>
                            </div>
                        </div>


                    </form>
                </div>
                <script src="pikaday.js"></script>
                <script>
                    var picker = new Pikaday({ field:
document.getElementById('datepicker') });
                </script>
            </body>
@endsection
```

```
@layout('layo
uts.main')
```

```
@section('extracss')
@endsection


@section('title')
    {{ show.Show_date }}
    @if(old('notification'))
```

```
            <div class="alert alert-danger">
                {{ old('notification') }}
            </div>
        @endif
    @endsection


    @section('content')
        <div class="future_shows color1">


            <form action="{{ route('ShowController.insert_ticket') }}"
    method="POST" class="color3">
                {{ csrfField() }}


                <div class="show_input">
                    <label for="Name">Name:</label>
                    <input type="text" name="Name">
                </div>


                <div class="show_input">
                    <label for="Phone_Number">Phone Number:</label>
                    <input type="text" name="Phone_Number">
                </div>


                <div class="show_input">
                    <label for="Seat_type">Seat Type:</label>
                    <input type="text" name="Seat_type">
                </div>


                <div class="show_input">
                    <label for="Seats_rsv">Seat Reserved:</label>
```

```
                    <input type="text" name="Seats_rsv">
                </div>


                <div id="submit_container">
                    <button type="submit">Submit</button>
                </div>


                <div class="custom_button color2">
                    <btn id="myBtn">View Seating Chart</btn>
                    <div id="myModal" class="modal">
                    <span class="close">&times;</span>
                    <img class="modal-content" id="myImg"
src="/VCPSeating.svg" alt="Seating Chart">
                    <div id="caption"></div>
                    </div>



                <script>
                    var modal = document.getElementById("myModal");

                    var btn = document.getElementById("myBtn");
                    var modalImg = document.getElementById("img01");
                    var captionText =
document.getElementById("caption");

                    btn.onclick = function(){
                      modal.style.display = "block";
                      modalImg.src = this.src;
                      captionText.innerHTML = this.alt;
                    }
```

```
                var span =
document.getElementsByClassName("close")[0];

                span.onclick = function() {
                  modal.style.display = "none";
                }
                </script>


            </div>




        </form>
    </div>


    <div class="vertical_space"></div>
    <h1>Current Tickets:</h1>


    <div class="show_display">
        @each(seating_chart in seating_charts)
            <div class="1 left_color">
                <div class="database_entry" id="add_ticket_db">
                    Name: {{seating_chart.Name}}<br>
                    Phone Number:
{{seating_chart.Phone_Number}}<br>
                    Seat_Type: {{seating_chart.Seat_type}}<br>
                    Seat Reserved: {{seating_chart.Seats_rsv}}<br>
                </div>
            </div>


            <div class="2">
```

```
                    <div class="custom_button color1">
                        <a
href="/add-ticket/edit-ticket/{{seating_chart.id}}" class =
"btn">Edit Ticket</a>
                    </div>
                    <div class="custom_button color2">
                        <a
href="/add-ticket/delete/{{seating_chart.id}}" class =
"btn">Delete Ticket</a>
                    </div>
                </div>


            @endeach
        </div>
@endsection
```

```
@layout('layout
s.main')
```

```
  @section('extracss')
  @endsection


  @section('title')
     Edit Shows
  @endsection


  @section('content')
     <div class="future_shows color1">
         <form action="{{ route('ShowController.update', { id:
  show.id}) }}" method="POST">
```

```
                    {{ csrfField() }}


                    <div id="show_input_container" class="color3">
                        <div class="show_input">
                            <label for="Show_title">Show Title:</label>
                            <input type="text" name="Show_title"
value="{{ show.Show_title }}">
                        </div>
                        <div class="show_input">
                            <label for="Show_date">Show Date:</label>
                            <input type="text" name="Show_date" value="{{
show.Show_date }}">
                        </div>


                        <div id="submit_container" class="color3">
                            <button type="submit">Submit</button>
                        </div>
                    </div>


                </form>
            </div>
        @endsection
```

```
@layout('la
youts.main'
)


    @section('extracss')
    @endsection
```

```
@section('title')
	{{ seating_chart.Name }}
@endsection


@section('content')
	<div class="future_shows color1">
		<form action="{{
route('SeatingChartController.update', { id: seating_chart.id}) }}"
method="POST" class="color3">
			{{ csrfField() }}


			<div class="show_input">
				<label for="Name">Name:</label>
				<input type="text" name="Name"
value="{{ seating_chart.Name }}">
			</div>
			<div class="show_input">
				<label for="Phone_Number">Phone
Number:</label>
				<input type="text" name="Phone_Number"
value="{{ seating_chart.Phone_Number }}">
			</div>
			<div class="show_input">
				<label for="Seat_type">Seat
Type:</label>
				<input type="text" name="Seat_type"
value="{{ seating_chart.Seat_type }}">
			</div>
			<div class="show_input">
				<label for="Seats_rsv">Seat
Reserved:</label>
				<input type="text" name="Seats_rsv"
value="{{ seating_chart.Seats_rsv }}">
			</div>
```

```
                                    <div id="submit_container">
                                    <button type="submit">Submit</button>
                                    </div>


                            </form>
                    </div>
            @endsection
```

```
@layout('layo
uts.main')


            @section('extracss')
            @endsection


            @section('title')
               Future Shows
            @endsection


            @section('content')
               <div class="show_display">
                   @each(show in shows)
                       <div class="database_entry">


                           <div class="left left_color">
                               <div class="left_container">
                                   <a href="/future-shows/{{show.id}}">
                                       Name: {{show.Show_title}}<br>
```

```
                                Date: {{show.Show_date}}
                            </a>
                            <div class="form_container">
                                <form action=" {{
route('ShowController.isPast', { id: show.id}) }}" method="POST">
                                    {{ csrfField() }}
                                    <button type="submit">
                                        Completed
                                    </button>
                                </form>
                            </div>
                        </div>
                    </div>


                    <div class="right" id="right_future_shows">
                        <div class="custom_button color1">
                            <a
href="/future-shows/edit-show/{{show.id}}" class="button">Edit</a>
                        </div>
                        <div class="custom_button color2">
                            <a href="/future-shows/delete/{{show.id}}"
class="button">Delete</a>
                        </div>
                        <div class="custom_button color3">
                            <a href="/print-tickets/{{show.id}}"
class="button">Print</a>
                        </div>
                    </div>


                    <div class="clear"></div>


                </div>
            @endeach
```

```
    </div>
@endsection
```

```
@layout('layouts.main')

@section('extracss')
@endsection


@section('title')
   Previous Shows
@endsection


@section('content')
   <div class="show_display">
       @each(show in shows)
           <div class="database_entry">


               <div class="left left_color">
                   <div class="previous_show_data"
                       <a href="/future-shows/{{show.id}}">
                           Name: {{show.Show_title}}<br>Date:
{{show.Show_date}}
                       </a>
                   </div>
                   <div class="undo">
                       <form action=" {{
route('ShowController.oops', { id: show.id}) }}" method="POST">
                       {{ csrfField() }}
```

```
                                    <button type="submit">
                                        Send back to Future Shows
                                    </button>
                                </form>
                            </div>
                        </div>


                        <div class="right">
                            <div class="custom_button color1">
                                <a
                href="/future-shows/edit-show/{{show.id}}" class ="btn">Edit</a>
                            </div>
                            <div class="custom_button color2">
                                <a href="/future-shows/delete/{{show.id}}"
                class ="btn">Delete</a>
                            </div>
                        </div>


                        <div class="clear"></div>


                    </div>
                @endeach
            </div>
        @endsection
```

```
<!DOCTYPE
html>
        <html lang="en">
```

```
<head>
   <meta charset="UTF-8" />
   <meta name="viewport" content="width=device-width,
initial-scale=1.0">
   <meta http-equiv="X-UA-Compatible" content="ie=edge">
   {{ style('reset') }}
   {{ style('print-style') }}


   <title>
       Print Preview
   </title>
</head>


<body>
   <div class="no-print">
       <h1>Preview of Ticket to be Printed:</h1>
   </div>


   <div class="print-container">
       <div class="ticket">
           <div class="ticket-element">
               Show:{{show.Show_title}}
           </div>
           <div class="ticket-element">
               Date: {{seating_chart.Show}}<br>
           </div>
           <br>
           <div class="ticket-element">
               Name: {{seating_chart.Name}}<br>
           </div>
           <div class="ticket-element">
```

```
                        Seat Type: {{seating_chart.Seat_type}}<br>
                    </div>
                    <div class="ticket-element">
                        Seat Reserved: {{seating_chart.Seats_rsv}}<br>
                    </div>
                </div>


                <div class="no-print">
                    Click "print" if the ticket information displayed within
            the pink border is correct<br>
                    <a href="javascript:window.print()" class = "btn">Print</a>
                </div>
            </div>
            </body>


            </html>
```

```
@layout('layouts.ma
in')

                @section('extracss')
                @endsection


                @section('title')
                   Print Tickets
                @endsection


                @section('content')
```

```
                    <h1>Click on a show's title to select tickets to
                print</h1>
                    <div class="show_display">
                        @each(show in shows)
                            <div class="database_entry" id="print_tickets_db">
                                <a href="/print-tickets/{{show.id}}">
                                    Name: {{show.Show_title}}<br>Date:
                {{show.Show_date}}
                                </a>
                            </div>
                        @endeach
                    </div>
                @endsection
```

```
@layout('layou
ts.main')
```

```
@section('extracss')
@endsection


@section('title')
   Print Tickets
@endsection


@section('content')
   <div class="picture">
       <h1>Displaying Seating Chart for {{ show.Show_date }}</h1>
       <h1>Show Title: {{ show.Show_title }}</h1>
   </div>
```

```
                    <div class="show_display">
                        @each(seating_chart in seating_charts)
                            <div class="database_entry" id="print_ticket_bg_color">
                                Name: {{seating_chart.Name}}<br>
                                Phone Number: {{seating_chart.Phone_Number}}<br>
                                Seat Type: {{seating_chart.Seat_type}}<br>
                                Seat Reserved: {{seating_chart.Seats_rsv}}<br>
                            </div>
                            <div class="links color2">
                                <a href="/print-tickets/print/{{seating_chart.id}}"
class = "btn">Print This Ticket</a>
                            </div>
                        @endeach
                    </div>
                @endsection
```

```
@layout('layouts.m
ain')
```

```
                    @section('extracss')
                    @endsection


                    @section('title')
                    @endsection


                    @section('content')
                        {{ show.Show_title }}
                        {{ show.Show_date }}
                        {{ seating_chart.Name }}
```

```
                    {{ seating_chart.Seat_type }}
                    {{ seating_chart.Seats_rsv }}
                    <input type="button" value="Print this page"
                onClick="window.print()">
                @endsection
```

```
@layout('layouts.
main')

                @section('extracss')
                @endsection


                @section('title')
                   Welcome
                @endsection


                @section('content')
                   <div class="logo">
                   </div>


                   <div class="subtitle">
                       <p>To view past shows click "Previous Shows"</p>
                       <p>To view future scheduled shows click "Future
                Shows"</p>
                       <p>To add a new show to the schedule click "Add
                Shows"</p>
                       <p>To print the tickets for a particular show click
                "Print Tickets"</P>
                   </div>
                @endsection
```

Start

```
'use
strict'


/*
|--------------------------------------------------------------------
-------
| Providers
|--------------------------------------------------------------------
-------
|
| Providers are building blocks for your Adonis app. Anytime you
install
| a new Adonis specific package, chances are you will register the
| provider here.
|
*/
const providers = [
 '@adonisjs/framework/providers/AppProvider',
 '@adonisjs/framework/providers/ViewProvider',
 '@adonisjs/lucid/providers/LucidProvider',
 '@adonisjs/bodyparser/providers/BodyParserProvider',
 '@adonisjs/cors/providers/CorsProvider',
 '@adonisjs/shield/providers/ShieldProvider',
 '@adonisjs/session/providers/SessionProvider',
 '@adonisjs/auth/providers/AuthProvider',
 '@adonisjs/validator/providers/ValidatorProvider'
]


/*
```

```
|--------------------------------------------------------------------
-------
| Ace Providers
|--------------------------------------------------------------------
-------
|
| Ace providers are required only when running ace commands. For
example
| Providers for migrations, tests etc.
|
*/
const aceProviders = [
 '@adonisjs/lucid/providers/MigrationsProvider'
]



/*
|--------------------------------------------------------------------
-------
| Aliases
|--------------------------------------------------------------------
-------
|
| Aliases are short unique names for IoC container bindings. You are
free
| to create your own aliases.
|
| For example:
|   { Route: 'Adonis/Src/Route' }
|
*/
const aliases = {}



/*
|--------------------------------------------------------------------
-------
```

```
| Commands
|--------------------------------------------------------------------------
-------
|
| Here you store ace commands for your package
|
*/
const commands = []


module.exports = { providers, aceProviders, aliases, commands }
```

```
'use
strict'


/** @type {import('@adonisjs/framework/src/Server')} */
const Server = use('Server')


/*
|--------------------------------------------------------------------------
-------
| Global Middleware
|--------------------------------------------------------------------------
-------
|
| Global middleware are executed on each http request only when the
routes
| match.
|
*/
const globalMiddleware = [
```

```
 'Adonis/Middleware/BodyParser',
 'Adonis/Middleware/Session',
 'Adonis/Middleware/Shield',
 'Adonis/Middleware/AuthInit',
 'App/Middleware/ConvertEmptyStringsToNull',
]



/*
|--------------------------------------------------------------------
-------
| Named Middleware
|--------------------------------------------------------------------
-------
|
| Named middleware is key/value object to conditionally add
middleware on
| specific routes or group of routes.
|
| // define
| {
|   auth: 'Adonis/Middleware/Auth'
| }
|
| // use
| Route.get().middleware('auth')
|
*/
const namedMiddleware = {
 auth: 'Adonis/Middleware/Auth',
 guest: 'Adonis/Middleware/AllowGuestOnly'
}
```

```
/*
|--------------------------------------------------------------------------
| Server Middleware
|--------------------------------------------------------------------------
|
| Server level middleware are executed even when route for a given URL is
| not registered. Features like `static assets` and `cors` needs better
| control over request lifecycle.
|
*/
const serverMiddleware = [
 'Adonis/Middleware/Static',
 'Adonis/Middleware/Cors'
]


Server
 .registerGlobal(globalMiddleware)
 .registerNamed(namedMiddleware)
 .use(serverMiddleware)
```

```
'use strict'


/*
|--------------------------------------------------------------------------
| Routes
```

```
|--------------------------------------------------------------------
-------
|
| Http routes are entry points to your web application. You can
create
| routes for different URL's and bind Controller actions to them.
|
| A complete guide on routing is available here.
| http://adonisjs.com/docs/4.1/routing
|
*/


/** @type {typeof import('@adonisjs/framework/src/Route/Manager')}
*/
const Route = use('Route')


// Render each view
Route.on('/').render('welcome')
Route.on('/add-show').render('add-show')


Route.get('/future-shows', 'ShowController.index')
Route.get('/print-tickets', 'ShowController.print_display')
Route.get('/print-tickets/:id', 'ShowController.print_tickets')
Route.get('/print-tickets/print/:id', 'ShowController.ticket')
Route.get('/previous-shows', 'ShowController.pastIndex')
Route.get('/future-shows/delete/:id', 'ShowController.delete')
Route.get('/future-shows/edit-show/:id', 'ShowController.edit')
Route.get('/add-ticket/edit-ticket/:id',
'SeatingChartController.edit')
Route.get('/add-ticket/delete/:id', 'SeatingChartController.delete')
Route.get('/future-shows/:id', 'ShowController.details')
```

```
Route.post('/future-shows/update/:id', 'ShowController.update')

Route.post('/add-ticket/update/:id',
'SeatingChartController.update')
Route.post('/add-show', 'ShowController.create')

Route.post('/future-shows', 'ShowController.insert_ticket')

Route.post('/future-Shows/isPast/:id', 'ShowController.isPast')

Route.post('/previous-shows/oops/:id', 'ShowController.oops')
```

Github - https://github.com/PeytonEllis/ESOF/tree/main

Site - esof423.cs.montana.edu:4001

Section 2: Teamwork

With the class being an Agile development class, planning for the application was done collectively via WebEx breakout rooms in a scrum environment. Outside of class communication was done through a shared messaging platform on a private server in Discord. Github was used for collaboration. Individual work was split up based on each member's strengths.

- Member 1 - Specialized in the backend of the web application and worked on items such as database creation, database CRUD operations, insertion validations, and overall application security. It is estimated that Member 1 has spent 40% of the total group time on the project.

- Member 2 - Focused on the interface and graphical design of the website while also taking on the less quantifiable and more miscellaneous jobs like GitHub management and writing. It is estimated that Member 2 has spent 25% of the total group time on the project.

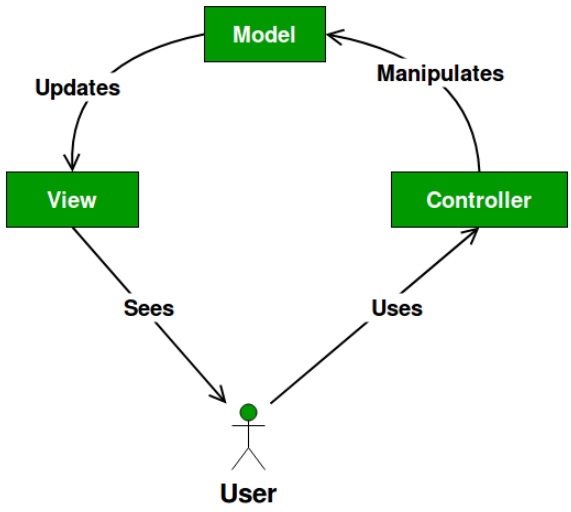
- Member 3 - Contributed to the frontend work in the user interface and focused on user experience. They specialized in items such as Ticket printing and the seating chart, as well as all CSS. It is estimated that Member 3 has spent 35% of the total group time working on the project.

Full backlog and burndown chart can be found here.

Project repository can be found here.

Section 3: Design Pattern

The design pattern utilized in the project was the Model-View-Controller (MVC) design pattern. The MVC design pattern is an architectural pattern and exists throughout the whole backend of the application. The MVC design pattern divides the program logic into three elements. The model sits between the view and the controller and it is the dynamic data structure. It directly takes care of data management and application logic. The view is used to present the data and allows the user to manipulate the data. The controller either denies or accepts the input and executes the correct action to send to the model.



We used AdonisJs as a backend framework for our application and AdonisJs allowed for easy implementation of the MVC design pattern along with the Adonis Router. The framework had a CLI command tool that would create the classes as needed. This made it very convenient to use the MVC pattern.

Section 4: Technical Writing

Application Technical Document

Mason Dinardi, Nathan Rubino, Peyton Ellis

ACM Reference Format:
Mason Dinardi, Nathan Rubino, Peyton Ellis. 2021. Ticketing Software Technical Document. In *proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA

1 INTRODUCTION

The following technical report details the process of building a ticketing software for the Virginia City Players theatre. The inspiration and motivation to attempt and complete this project came from our computer science capstone course at Montana State University. The goal of this project was to create a fully functional and feature complete web application that serves as a ticketing software for the Virginia City Players. This software would take care of actions such as but not restricted to adding and editing shows, adding and editing individual tickets for said shows, viewing seating charts for shows, and printing individual tickets for selected shows. We will start with a short background for the project specifications, followed by an in-depth description of the methods of the implementation of the web application. Finally, we will have a short conclusion and future work section.

---

2 BACKGROUND

The web application that was developed was requested by the owner of the Virginia City Players as a means to replace the current ticketing system that is used. The current ticketing system is completely manual, using pen and paper to fill out seating charts and tickets. The proposed solution that we came up with consisted of a web application that could run on a local server. This way the application is more secure, and no

outside manipulation can be done. The requirements for the application were to: (1) automate the process for tracking available seating for a performance, and (2) format nightly seating data for print on demand tickets. The application would require: (1) database of shows, seating, and guest information and (2) UI interaction with database and formatted ticket data to print.

3 METHODS AND DISCUSSIONS

3.1 Tools and Setup

MySQL is the database tool that we will be using for this project. AdonisJs is the backend framework that we will be using. AdonisJs is a NodeJs framework and is built on top of NodeJs so NodeJs will also have to be installed on our machines. Node Package Manager is also required for the project structure. There are a few AdonisJs packages that will also have to be installed. These packages include @adonisjs/cli for the Adonis command line user interface and @adonisjs/validator for database insertion validations. The application takes advantage of the MVC architectural design pattern conveniently implemented by the AdonisJs framework. We followed the installation instructions for each of the dependencies and the project setup was complete.

We will also use GitHub as a version control solution for this project. These resources were installed locally on all of our machines and will allow for different operating systems and machines to still collaborate together.

3.2 Database

MySQL database management service is used for the web application. There are two tables that the application uses. There is a “shows” table and a “seating_charts” table.

The shows table will hold all data for a given show. The columns contained in shows consist of the following attributes: id, Show_title, Show_date, isPast, created_at, and updated_at. Show_title and Show_date are both of string datatypes, id is integer datatype, isPast is Boolean datatype, created_at and updated_at are both dateTime datatypes. Show_date is unique.

The seating_charts table holds all data for a show’s seating arrangement. The columns contained in seating_charts consist of id, Name, Phone_Number, Seat_type, Seats_rsv, Show, created_at, and updated_at. Name, Phone_Number, Seat_type, Seats_rsv, and Show are all of the string datatype, id is integer datatype, and created_at and updated_at are both of dateTime datatype. Show is a foreign key reference to Show_date in the shows table. This way data from all shows can be kept track of. For example, if a user

wanted to see which seats are taken up for a particular show, they would be able to do that.

The creation and upkeep of these tables is done dynamically via migrations using Adonis CLI commands. All CRUD operations and queries are taken care of in the controllers.

3.3 Application

The MVC design pattern divides the program logic into three elements. The model sits between the view and the controller and it is the dynamic data structure. It directly takes care of data management and application logic. The view is used to present the data and allows the user to manipulate the data. The controller either denies or accepts the input and executes the correct action to send to the model.

There are two models used in this application: SeatingChart.js and Show.js. Both of these models inherit from a parent model class and no further methods were needed for implementation.

There are two controllers used in this application: SeatingChartController.js and ShowController.js. These controllers are where the bulk of the programming happened. They take care of all CRUD functions, as well as edits, indexing, querying, displaying views and more. The ShowController handles the majority of the applications operations and input.

There are 12 views used in this application. Each view displays different information, and they all inherit from a main.edge template file that takes care of the standard formatting for the web application. These views combined with three stylesheets are what the frontend is composed of.

The three stylesheets are CSS and take care of all web formatting for the .edge files. There is one stylesheet that is used only for printing the tickets, this way unwanted noise is removed from the print layout.

4 CONCLUSIONS AND FUTURE WORK

To conclude our technical report, we would like to examine some aspects of the application that could benefit from future work. With the instructions and requirements given, our application accomplishes the goals of the project, being that we (1) automate the process for tracking available seating for a performance, and (2) format nightly seating data for print on demand tickets. However, like most web applications, there is plenty of room for improvement.

The major candidate for future work is utilizing CSS templates and moving our frontend to a framework. This would improve usability and overall appearance of the application greatly.

Another candidate for future improvement is adding user

functionality. Adding this would allow us to host this site publicly without allowing users with nefarious intentions to manipulate the data. It would also allow users to make their own accounts and have the possibility of administrator privileges to the owner of the site.

Further refinement and optimization of the application would be beneficial to the usability of the website; however, our implementation still captures all features and requirements necessary to complete the tasks at hand.

To view more official dev/user documentation of the application navigate here.

Section 5: UML

Diagram 1

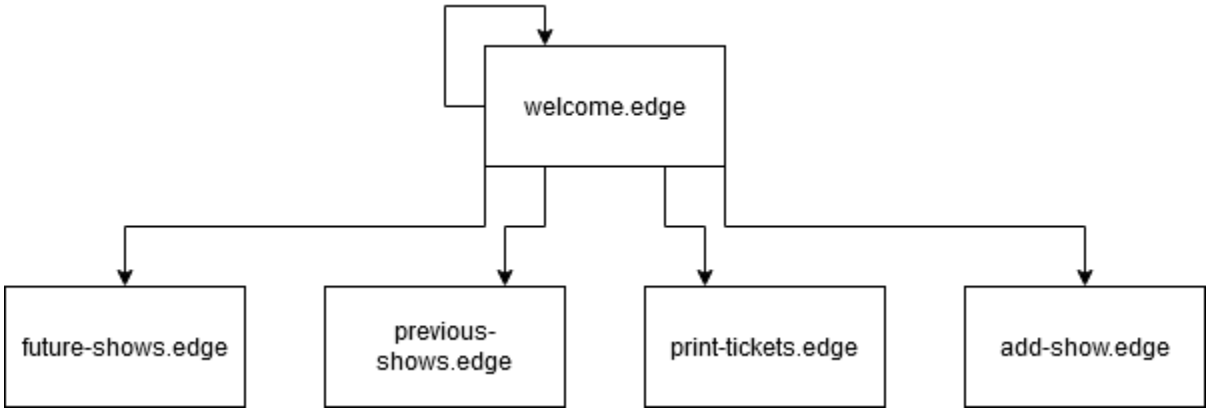
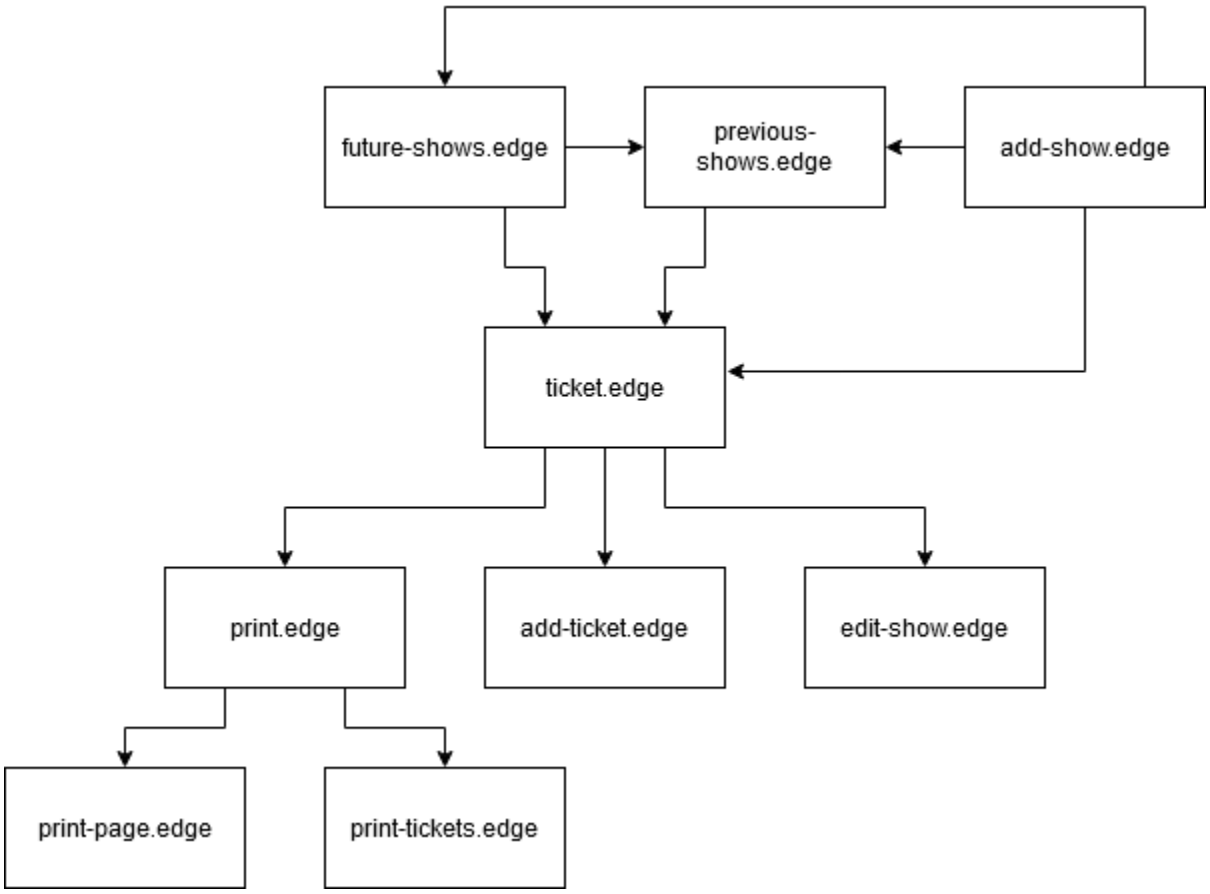
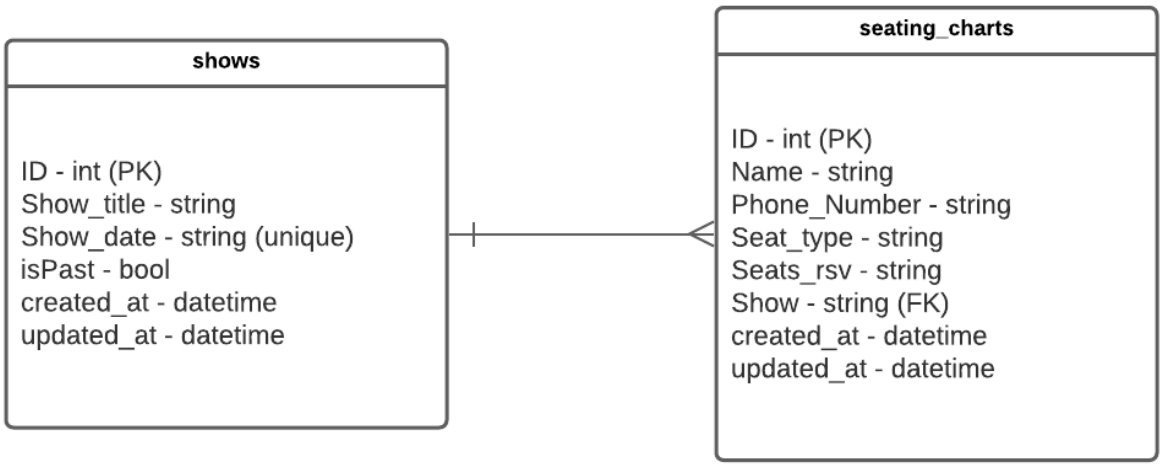


Diagram 2



Database Model



Section 6: Design Trade-Offs

The group originally intended to have a clickable seating chart for the ticketing system. However due to the difficulty of implementing such a system from scratch, as well as time restrictions due to other items also needing to be completed the idea was shelved. In its place was a text-based system that allowed the user to manually input the customers name, seat location, and any extraneously relevant information like discounts or a disability.

Section 7: Software Development Life Cycle Model

The group used the SCRUM life cycle model. The group met every Monday and Friday to discuss our current progress on the project, as well as any obstacles members faced. There were a total of seven sprints over the semester, all roughly the same in terms of time required and difficulty. There was a point where the week one sprint had incomplete items and it was with the help of the SCRUM model that the group was able to get back on track.

Every other Friday had the group present its current progress to either Professor DeFrance or Anna Jinneman. From there the group would reflect upon the feedback over the weekend, and regroup on the following Monday to plan out the backlog for the upcoming two weeks. In addition to the backlog was a burndown chart that kept track of the groups progress on a meeting-to-meeting basis. Overall, the SCRUM model had a great impact upon the group, and was a key component in keeping each individual member on track.