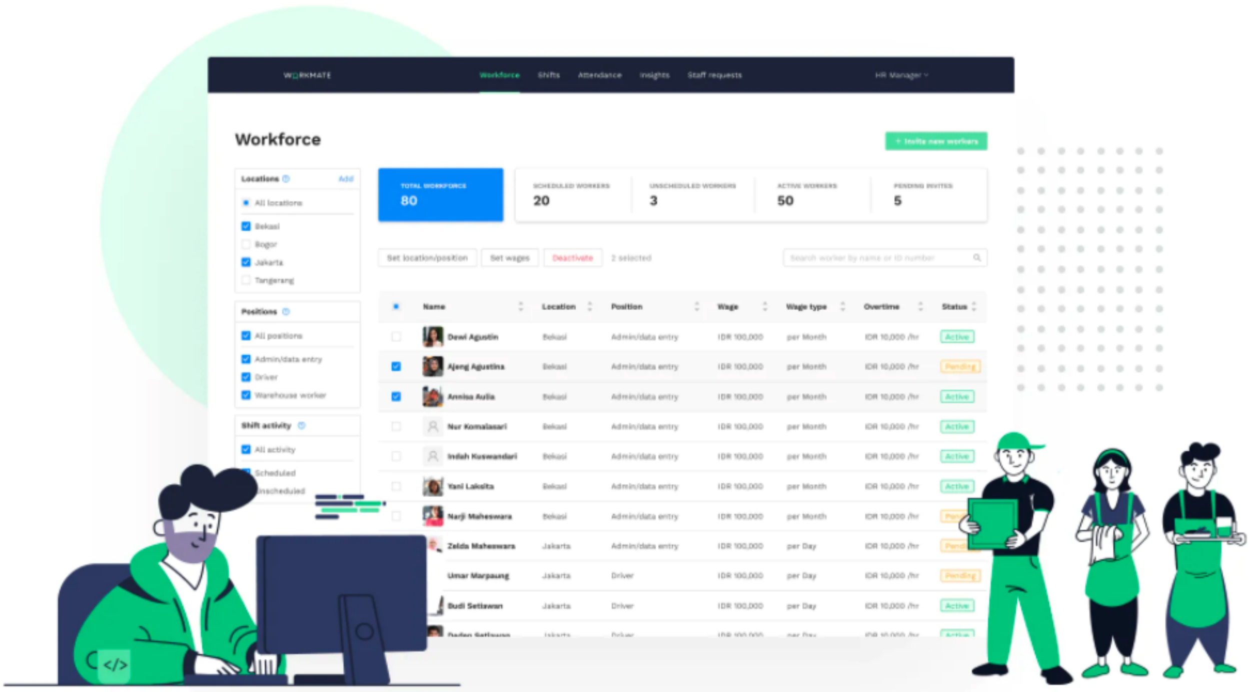
HÖGSKOLAN VÄST

University West,Trollhattan
School of Business, Economics and IT
Division of Informatics

FSU600 - PROJECT REPORT,FALL 2021
Prudent Digital planner App

December 8, 2021



Author: *Kanniga Lakshmi Jagadeesan – kanniga-lakshmi.jagadeesan@student.hv.se*
Supervisor: *Abdulghafour Mohammad - abdulghafour.mohammad@hv.se*

Contents

1 Project Background

A *Full Stack* application combined with *Agile* methodology aims at setting up an environment to develop software based on principles from the agile manifesto. This can achieved with the help of Continuous Integration and Continuous Delivery which act as pillars for *DevOps*.The agile methodology encourages iterative and incremental development through a design-code-test cycle.Agile principles has been a backbone to this project to develop a Full Stack application.Since, this is an individual project,not all but atleast few of the agile principles will be followed throughout the development phase of this project.Digitalization has proved to be a major trend with the help of Informatics covering a wide range of areas.

Traditional Waterfall model involves development task with several teams weighing down the delivery lifecycle.This has been overcome by Agile methodologies which involves development task and DevOps simultaneously at a constant pace. The software that will be delivered in the end of this project is a 'Planner application' for an amusement park with the purpose of providing an overview of all support personnel in the park along with their assigned schedule and other additional features like leaves,departments,etc..,Although there has been previously developed apps in a similar fashion, this app is unique in a way that, the application will have a colourful theme in order to mimic the colours of the amusement park.Furthermore, a simple layout will be implemented for usability and readability purposes.

2 Development task

As like any application, the ultimate purpose of this project is to deliver a minimum viable product with basic functionalities at the UI and backend level to customer. At the same time, the product should focus on minimum bugs as well as good performance with *'Customer Satisfaction'* as high priority so as to obtain the end result, a *Happy Customer*.

2.1 Technical Stack

This application is developed using *Java Script* and *MongoDB*, a non-relational database on an Agile framework.

Frontend	Backend	Database	API Testing	IDE
Vue.js	Node.js	MongoDB	Postman	Visual Studio Code

Table 1: Technical stack used in Project

3 Project Description

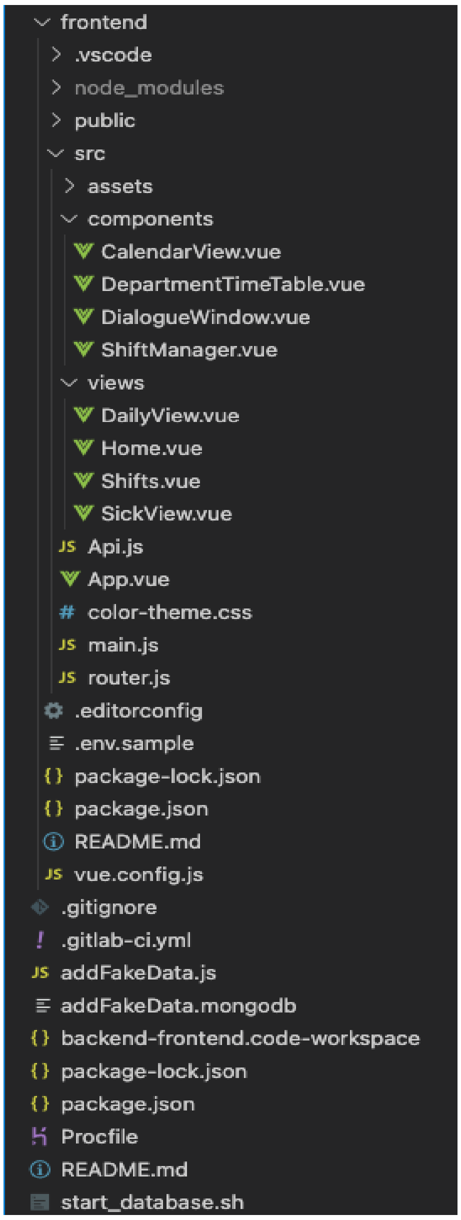
The entire project development was accomplished by division of task.For better understanding,the task division has been given in terms of user stories planned for each week and activities involved with the time spent per task in Table 2.This has further been used to calculate the throughput, performance for better analysis.

Table 2: Project activity logs

User stories	**Activity log**	**Time spent** *(in hrs)*
As a planner,I want to	**Week 1**	
• Display the overview in a calendar view. • Create a shift for an employee. (Only UI part)	• Setup frontend. • Setup backend. • Create calendar component, input form. • CI/CD pipeline.	15
As a planner,I want	**Week 2**	
• Database to store information. • To Manage two types of schedules: Normal and Emergency. • To Modify shifts for an employee. • Overview of shifts on a specific day.	• Create database and add fake data. • Update and Delete operation on Shifts. • Connect frontend with backend. • Detailed Calendar view(CSS), Emergency shifts and load calendar items.	20
As a planner,I want to	**Week 3**	
• Display total hours for each staff at the right side of the assigned shift in daily view and weekly view with assigned staffs and hours per week.	• Show real calendar items and distinction of emergency and normal shifts. • Queries to provide number of assigned staff and total hours for a date and week. • CRUD operations on Personnel and Department.	10
As a planner,I want	**Week 4**	
• Call-in sick functionality. • To check conflict of staffs' time.	• Add "isSick" boolean to db, backend etc.,update all calculations,Create view with "Call- in sick" functionality. • Display shift differently for Sick staff. • Check overlapping date and time during the shift assignment for a staff and throw error.	14

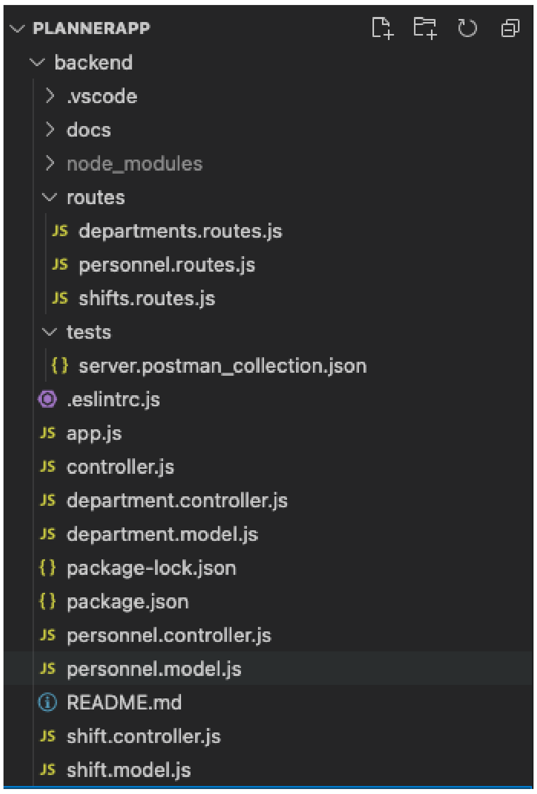
3.1 Project structure

For an easy configuration, better understanding and to establish proper communication between the frontend and backend,the project structure has been divided into two main folders as *frontend* and *backend*.Frontend contains most of the UI sections which comprises of Views and components that are present inside each view.*'App.Vue'* is the main View which the user can see on entering the application, each views are further called after each operation. Backend contains all the code necessary to connect with the database and pass the information from the frontend to backend.*'app.js'* is the main file to establish connection with the database and which later lets the user to perform *CRUD* operations in the application and the data gets updated in the database via each js files related to that particular *Collections*(similar to tables in a relational database.)



```
frontend
  > .vscode
  > node_modules
  > public
  src
    > assets
    components
      CalendarView.vue
      DepartmentTimeTable.vue
      DialogueWindow.vue
      ShiftManager.vue
    views
      DailyView.vue
      Home.vue
      Shifts.vue
      SickView.vue
    Api.js
    App.vue
    color-theme.css
    main.js
    router.js
  .editorconfig
  .env.sample
  package-lock.json
  package.json
  README.md
  vue.config.js
.gitignore
.gitlab-ci.yml
addFakeData.js
addFakeData.mongodb
backend-frontend.code-workspace
package-lock.json
package.json
Procfile
README.md
start_database.sh
```

(a) Frontend



```
PLANNERAPP
  backend
    > .vscode
    > docs
    > node_modules
    routes
      departments.routes.js
      personnel.routes.js
      shifts.routes.js
    tests
      server.postman_collection.json
    .eslintrc.js
    app.js
    controller.js
    department.controller.js
    department.model.js
    package-lock.json
    package.json
    personnel.controller.js
    personnel.model.js
    README.md
    shift.controller.js
    shift.model.js
```

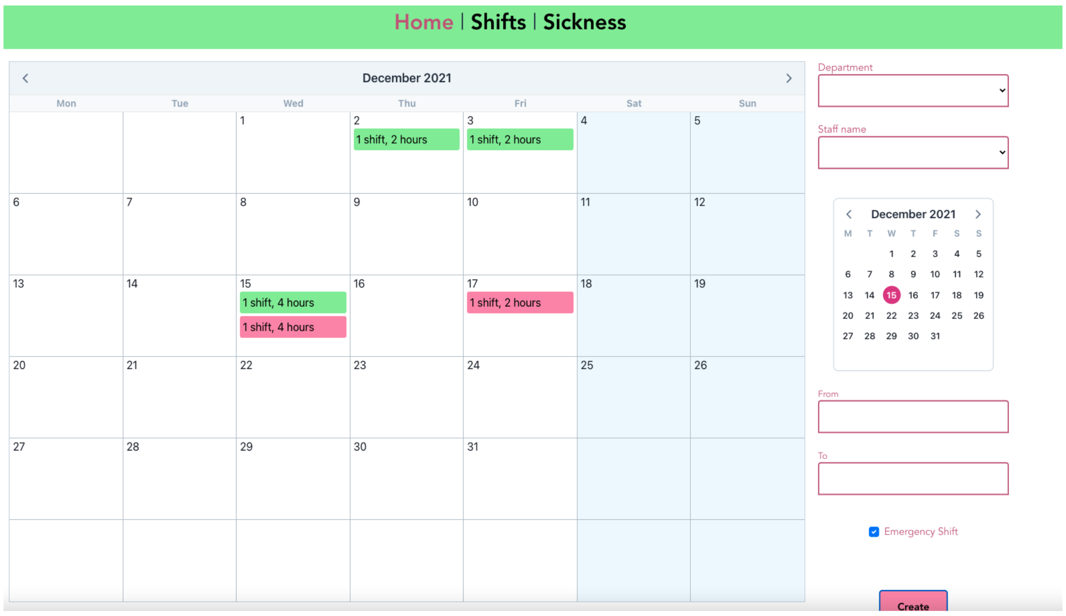
(b) Backend

Figure 1: Project structure

3.2 Project Outcome

The final planner application consists of four major components.

- The first view is the *'Home'* page as shown in figure 2,which provides an overview of the month with planned shifts and number of hours that is scheduled for that day. Additionally, new shifts can be added with selection of department, personnel and time/date of planner's choice, and in case a personnel needs to be assigned an emergency shift,the check box should be selected within this view.



December 2021

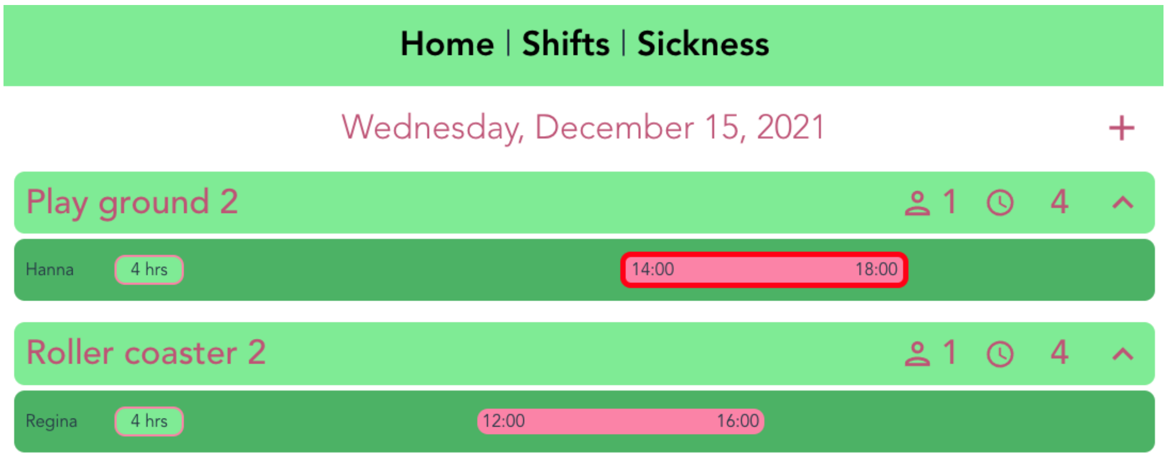
Mon	Tue	Wed	Thu	Fri	Sat	Sun
		1	2 1 shift, 2 hours	3 1 shift, 2 hours	4	5
6	7	8	9	10	11	12
13	14	15 1 shift, 4 hours 1 shift, 4 hours	16	17 1 shift, 2 hours	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

December 2021

M	T	W	T	F	S	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

Figure 2: Home page of Planner app

- The second component is the *'Daily'* view and the reason for this is because it provides an detailed view of that single date, such as which planned shift needs replacement in-case of a personnel getting sick, which hours of the day that are covered and by who, and lastly also how many personnel that are scheduled and their total hours as shown in figure 3.



Wednesday, December 15, 2021

Play ground 2 1 4

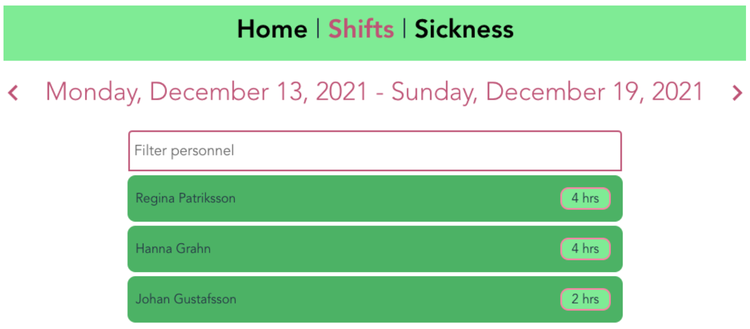
Hanna 4 hrs 14:00 18:00

Roller coaster 2 1 4

Regina 4 hrs 12:00 16:00

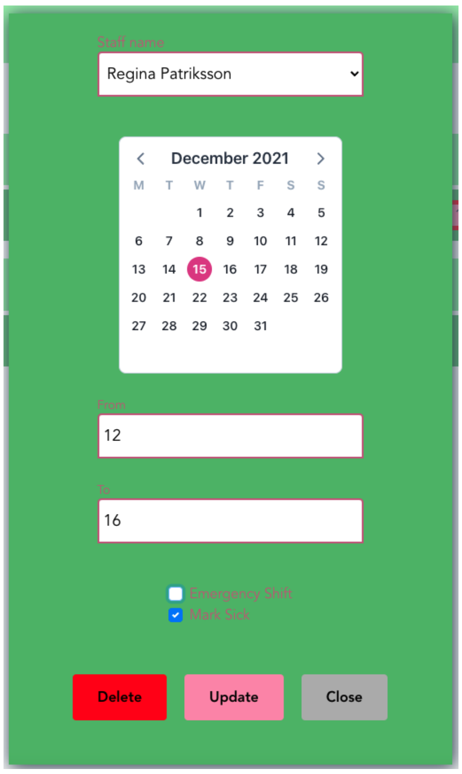
Figure 3: Daily Shifts view

- The third component is an *'Shift'* overview.This one contributes a list of all available personnel and how many hours they have scheduled as shown in Figure 4. Figure 5 shows the changes in the shift before and after update operation.The emergency shifts can be seen marked with a 'red' border and personnel marked as sick in 'yellow' color with a sick emoji.

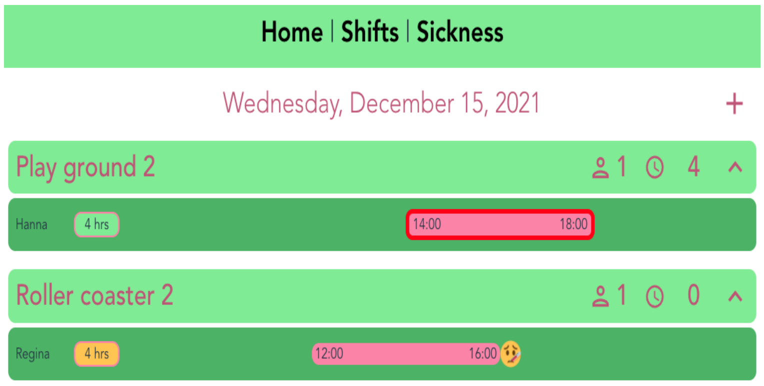


Regina Patriksson 4 hrs
Hanna Grahn 4 hrs
Johan Gustafsson 2 hrs

Figure 4: List of shifts scheduled



(a) Pre update of a shift



(b) Post update of a shift

Figure 5: CRUD operation on a Shift

- Last component is the *'Sick'* view which allows the personnel to call in sick when needed, which has the option to edit/delete their "planned" sick leave.Furthermore, this view also presents all personnel that are currently sick.

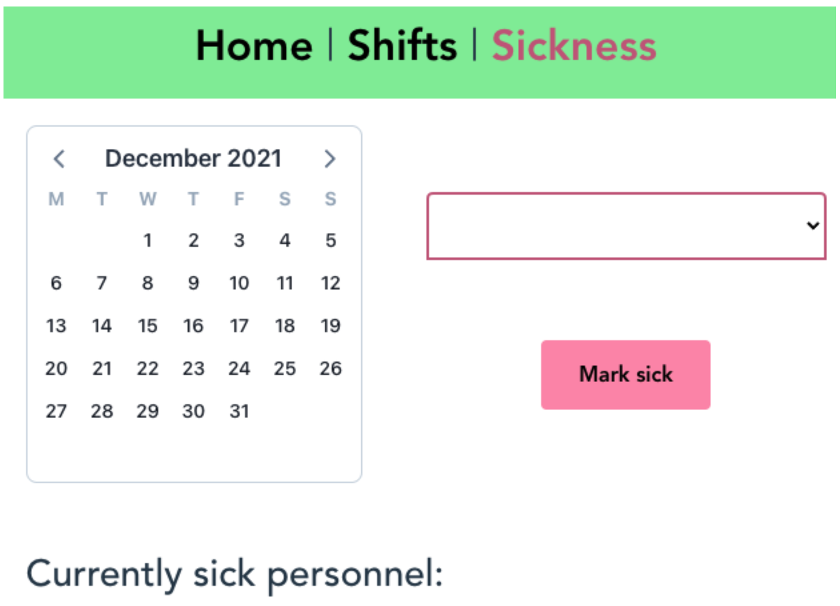


Figure 6: Sick view

4 Future Extensions

A tentative overview of User Stories that will be implemented as upcoming tasks is given below.

- Create *sign-in* authentication for planner.
- *Profile page* for Personnel.
- Automatic staff replacement suggestion on Personnel's absence.
- *'Career'* view to add job listings by planner.
- Screen size compatibility on multi-devices.
- Deploy the application to a *Cloud* platform.

5 Reflections

Every information from the lecture and the assignments in this Full Stack Developer course regularly gave new information about the technical elements that helped to implement in this project.This set a *good standard* way of working process which helped with task division and coordinate between the frontend and backend tasks.However,few *technical debts* had to be carried forward every week,one reason being time constraint and other reason is the absence of *'pair programming'*.In terms of *'Throughput'* and *'Performance'*, all the commitments that were planned for that particular week were fulfilled above the average line.One important criteria that could have been implemented but missed during the development is *'Test Driven Development'*, which called for *'Manual Testing'* that consumed more time. The conclusion that can be drawn is that changes are inevitable and that change management(DevOps) therefore has to work more closely with the development process.