
Cinnamon Documentation

Release 1.0

Alexander Ray

Nov 28, 2018

CONTENTS

1	project.controllers package	3
1.1	project.controllers.AddAccountController module	3
1.2	project.controllers.CreateReportController module	3
1.3	project.controllers.HomeController module	3
1.4	project.controllers.LoginController module	4
1.5	project.controllers.LogoutController module	4
1.6	project.controllers.SettingsController module	4
1.7	project.controllers.SignupController module	4
1.8	project.controllers.SpendingController module	5
1.9	project.controllers.forms module	5
2	project.models package	7
2.1	project.models.Account module	7
2.2	project.models.Address module	8
2.3	project.models.ReportGenerator module	8
2.4	project.models.ReportType module	9
2.5	project.models.SpendingHistory module	9
2.6	project.models.SpendingInstance module	9
2.7	project.models.SpendingInstanceFactory module	10
2.8	project.models.SpendingType module	10
2.9	project.models.SummaryGenerator module	11
2.10	project.models.SummaryType module	11
2.11	project.models.User module	11
2.12	project.models.UserInformation module	12
	Python Module Index	13
	Index	15

Contents:

PROJECT.CONTROLLERS PACKAGE

1.1 project.controllers.AddAccountController module

```
class project.controllers.AddAccountController.AddAccountController
    Bases: flask.views.View

    decorators = [<function login_required>]

    dispatch_request()
        Handler for add account controller. Takes data from account add form, creates new account with decorator
        classes

        Returns Template

    methods = ['GET', 'POST']
```

1.2 project.controllers.CreateReportController module

```
class project.controllers.CreateReportController.CreateReportController
    Bases: flask.views.View

    decorators = [<function login_required>]

    dispatch_request()
        Handler for create report. Takes filename data from form and generates report with user's current report
        generator

        Returns Template

    methods = ['GET', 'POST']
```

1.3 project.controllers.HomeController module

```
class project.controllers.HomeController.HomeController
    Bases: flask.views.View

    decorators = [<function login_required>]

    dispatch_request()
        Handler for requests to homepage. Retrieves data from model and passes to view

        Returns Template
```

1.4 project.controllers.LoginController module

```
class project.controllers.LoginController.LoginController
    Bases: flask.views.View

    dispatch_request()
        Login handler. Compares username and password to database, sends to homepage if approved

        Returns Template

    methods = ['GET', 'POST']
```

1.5 project.controllers.LogoutController module

```
class project.controllers.LogoutController.LogoutController
    Bases: flask.views.View

    dispatch_request()
        Handler for logout

        Returns Redirect to login page

    methods = ['GET', 'POST']
```

1.6 project.controllers.SettingsController module

```
class project.controllers.SettingsController.SettingsController
    Bases: flask.views.View

    decorators = [<function login_required>]

    dispatch_request()
        Settings handler. Populates default fields with existing data. Recreates report generator based on user
        choices, following strategy design pattern.

        Returns Template

    methods = ['GET', 'POST']
```

1.7 project.controllers.SignupController module

```
class project.controllers.SignupController.SignupController
    Bases: flask.views.View

    dispatch_request()
        Signup handler. Takes form data and creates a new user

        Returns Template

    methods = ['GET', 'POST']
```


1.8 project.controllers.SpendingController module

```
class project.controllers.SpendingController.SpendingController
    Bases: flask.views.View

    decorators = [<function login_required>]

    dispatch_request()
        Handler for adding spending instances. Populates default fields and drop down menus. Withdraws amount
        from specified account, creates a new spending instance.

        Returns Template

    methods = ['GET', 'POST']
```

1.9 project.controllers.forms module

```
class project.controllers.forms.AddAccountForm(*args, **kwargs)
    Bases: flask_wtf.form.Form

    account = <UnboundField(StringField, ('Account',), {'validators': [<wtforms.validators.InputIfTrue>]})>
    include_email = <UnboundField(BooleanField, ('Email Updates',), {})>
    include_savings = <UnboundField(BooleanField, ('Save by Rounding',), {})>

class project.controllers.forms.BaseAddressForm(*args, **kwargs)
    Bases: flask_wtf.form.Form

    city = <UnboundField(StringField, ('City',), {'validators': [<wtforms.validators.InputIfTrue>]})>
    state = <UnboundField(SelectField, ('State',), {'coerce': <class 'str'>, 'validators': [<wtforms.validators.InputIfTrue>]})>
    street_address = <UnboundField(StringField, ('Street Address',), {'validators': [<wtforms.validators.InputIfTrue>]})>
    zip = <UnboundField(StringField, ('Zip Code',), {'validators': [<wtforms.validators.InputIfTrue>]})>

class project.controllers.forms.BaseUserForm(*args, **kwargs)
    Bases: flask_wtf.form.Form

    email = <UnboundField(EmailField, ('Email',), {'validators': []})>
    password = <UnboundField(PasswordField, ('Password',), {'validators': [<wtforms.validators.InputIfTrue>]})>

class project.controllers.forms.CreateReportForm(*args, **kwargs)
    Bases: flask_wtf.form.Form

    filename = <UnboundField(StringField, ('Filename',), {'validators': [<wtforms.validators.InputIfTrue>]})>
    include_description = <UnboundField(BooleanField, ('Include Description',), {})>

class project.controllers.forms.LogSpendingForm(*args, **kwargs)
    Bases: flask_wtf.form.Form

    account = <UnboundField(SelectField, ('Account',), {'coerce': <class 'str'>, 'validators': [<wtforms.validators.InputIfTrue>]})>
    amount = <UnboundField(DecimalField, ('Amount',), {'validators': [<wtforms.validators.InputIfTrue>]})>
    date = <UnboundField(DateField, ('Date of Spending',), {'format': '%Y-%m-%d', 'default': datetime.datetime.now()})>
    description = <UnboundField(StringField, ('Description',), {'validators': [<wtforms.validators.InputIfTrue>]})>
    spending_type = <UnboundField(SelectField, ('Spending Type',), {'coerce': <class 'str'>, 'validators': [<wtforms.validators.InputIfTrue>]})>
```

```
class project.controllers.forms.SettingsForm(*args, **kwargs)
    Bases: project.controllers.forms.BaseAddressForm

    income = <UnboundField(IntegerField, ('Income',)), {'validators': [<wtforms.validators
    report_type = <UnboundField(SelectField, ('Report Type',)), {'coerce': <class 'str'>,
    summary_type = <UnboundField(SelectField, ('Summary Type',)), {'coerce': <class 'str'>

class project.controllers.forms.SignupForm(*args, **kwargs)
    Bases: project.controllers.forms.BaseUserForm, project.controllers.forms.
    BaseAddressForm

    income = <UnboundField(IntegerField, ('Income',)), {'validators': [<wtforms.validators
```

PROJECT.MODELS PACKAGE

2.1 project.models.Account module

```
class project.models.Account.AccountEmailDecorator (dec)
```

Bases: *project.models.Account.BaseAccountDecorator*

```
withdraw (amount)
```

Withdraw method for email decorator class. Calls other withdraw methods, then sends email with details

Parameters *amount* – Amount of money to withdraw

Returns Current total expenditure. Needed for decorator design pattern due to SQLAlchemy constraints

```
class project.models.Account.AccountSavingsDecorator (dec)
```

Bases: *project.models.Account.BaseAccountDecorator*

```
withdraw (amount)
```

Withdraw method for savings decorator. Calls other withdraw methods, adds to savings (rounded up to nearest dollar), and returns current sum of withdrawals.

Parameters *amount* – Amount of money to withdraw

Returns Current total expenditure. Needed for decorator design pattern due to SQLAlchemy constraints

```
class project.models.Account.BaseAccount (**kwargs)
```

Bases: *sqlalchemy.ext.declarative.api.Model*

```
expenditure
```

Getter for total expenditure

Returns Sum of all spending in this account

```
name
```

Getter for account name

Returns Name of account

```
withdraw (amount)
```

Abstract withdraw method, to be implemented by subclasses

Parameters *amount* – Amount to withdraw

```
class project.models.Account.BaseAccountDecorator (dec)
```

Bases: *project.models.Account.BaseAccount*

```
withdraw (amount)
```

Abstract withdraw method, to be implemented by subclasses

Parameters **amount** – Amount to withdraw

class `project.models.Account.ConcreteAccount` (*name*)

Bases: `project.models.Account.BaseAccount`

withdraw (*amount*)

Method for withdrawing money from an account

Parameters **amount** – Amount of money to withdraw

Returns Current total expenditure. Needed for decorator design pattern due to SQLAlchemy constraints

2.2 project.models.Address module

class `project.models.Address.Address` (*street_address, city, state, zip*)

Bases: `sqlalchemy.ext.declarative.api.Model`

city

Getter for city

Returns City

state

Getter for state

Returns State

street_address

Getter for street address

Returns Street address

zip

Getter for zip code

Returns Zip code

2.3 project.models.ReportGenerator module

class `project.models.ReportGenerator.CSVReportGenerator` (*filename*)

Bases: `project.models.ReportGenerator.ReportGenerator`

generate_report (*instances, include_description*)

CSV report generator, using Python's CSV module

Parameters

- **instances** – List of spending instance objects
- **include_description** – Toggle for including description

Returns Flask response object

class `project.models.ReportGenerator.JSONReportGenerator` (*filename*)

Bases: `project.models.ReportGenerator.ReportGenerator`

generate_report (*instances, include_description*)

JSON report generator, using Python's JSON module

Parameters

- **instances** – List of spending instance objects
- **include_description** – Toggle for including description

Returns Flask reponse object

class `project.models.ReportGenerator.ReportGenerator` (*filename*)

Bases: `sqlalchemy.ext.declarative.api.Model`

default_filename

Getter for default filename of report generator

Returns Default filename

generate_report (*instances, include_description*)

Abstract method to generate report, must be implemented by subclasses

Parameters

- **instances** – List of spending instance objects
- **include_description** – Boolean value to determine whether or not to include a description

2.4 project.models.ReportType module

class `project.models.ReportType.ReportType`

Bases: `enum.Enum`

An enumeration.

CSV = 'CSV'

JSON = 'JSON'

2.5 project.models.SpendingHistory module

class `project.models.SpendingHistory.SpendingHistory` (***kwargs*)

Bases: `sqlalchemy.ext.declarative.api.Model`

add_spending_instance (*instance*)

Method to add a spending instance to your spending history

Parameters **instance** – Instance to add

spending_instances

Getter for spending instances

Returns List of spending instances, sorted in reverse chronological order by date

2.6 project.models.SpendingInstance module

class `project.models.SpendingInstance.DiningSpendingInstance` (*amount, account, date, description*)

Bases: `project.models.SpendingInstance.SpendingInstance`

```
class project.models.SpendingInstance.RetailSpendingInstance (amount, account,  
                                                             date, description)  
    Bases: project.models.SpendingInstance.SpendingInstance  
class project.models.SpendingInstance.SpendingInstance (amount, account, date, de-  
                                                             scription)  
    Bases: sqlalchemy.ext.declarative.api.Model  
  
    account  
        Getter for account associated with spending instance  
        Returns Account  
  
    amount  
        Getter for spending instance amount  
        Returns Amount  
  
    date  
        Getter for spending instance date  
        Returns Date  
  
    description  
        Getter for description associated with spending instance  
        Returns Description  
  
    id  
        Getter for id  
        Returns Id
```

2.7 project.models.SpendingInstanceFactory module

```
class project.models.SpendingInstanceFactory.SpendingInstanceFactory  
    Bases: object  
  
    static factory_method (amount, account, date, description, instance_type)  
        Static factory method for simple factory design pattern, to create spending instances. Types defined in  
        SpendingType  
  
        Parameters  


- amount – Amount of instance
- account – Account object
- date – Date object
- description – Optional description
- instance_type – Type from SpendingType

  
        Returns Spending instance object
```

2.8 project.models.SpendingType module

```
class project.models.SpendingType.SpendingType  
    Bases: enum.Enum
```

An enumeration.

```
DINING = 'Dining'
RETAIL = 'Retail'
```

2.9 project.models.SummaryGenerator module

```
class project.models.SummaryGenerator.DateOrientedSummaryGenerator(**kwargs)
    Bases: project.models.SummaryGenerator.SummaryGenerator

class project.models.SummaryGenerator.SummaryGenerator(**kwargs)
    Bases: sqlalchemy.ext.declarative.api.Model

    summary_template_method()
        Template method for summary generator. Enforces ordering of summary generation using the template
        method design pattern

        Returns HTML-formatted string

class project.models.SummaryGenerator.TotalExpenditureSummaryGenerator(**kwargs)
    Bases: project.models.SummaryGenerator.SummaryGenerator
```

2.10 project.models.SummaryType module

```
class project.models.SummaryType.SummaryType
    Bases: enum.Enum

    An enumeration.

    DATE_ORIENTED = 'Date-oriented'

    TOTAL_EXPENDITURE = 'Total Expenditure'
```

2.11 project.models.User module

```
class project.models.User.User(email, password, info)
    Bases: sqlalchemy.ext.declarative.api.Model

    account_names
        Account names property to retrieve list of account names

        Returns List of account names associated with user

    accounts
        Getter for list of user accounts

        Returns List of user accounts

    email
        Getter for user email

        Returns User email

    get_account(name)
        Get user account by name
```

Parameters **name** – Name of user account

Returns Account object or null if not found

get_id()

flask-login method for id retrieval

Returns User id

information

Getter for user information object

Returns User information object

is_active()

flask-login method

Returns True

is_anonymous()

flask-login method

Returns False

is_authenticated()

flask-login method

Returns True

password

Getter for user password

Returns User password

report_generator

Getter for report generator object

Returns Report generator object

spending_history

Getter for user spending history

Returns Spending history object

summary_generator

Getter for summary generator object

Returns Summary generator object

2.12 project.models.UserInformation module

class `project.models.UserInformation.UserInformation(income, address)`

Bases: `sqlalchemy.ext.declarative.api.Model`

address

Getter for user address

Returns User address object

income

Getter for user income

Returns User income

PYTHON MODULE INDEX

p

- `project.controllers.AddAccountController,`
3
- `project.controllers.CreateReportController,`
3
- `project.controllers.forms,` 5
- `project.controllers.HomeController,` 3
- `project.controllers.LoginController,` 4
- `project.controllers.LogoutController,` 4
- `project.controllers.SettingsController,`
4
- `project.controllers.SignupController,` 4
- `project.controllers.SpendingController,`
5
- `project.models.Account,` 7
- `project.models.Address,` 8
- `project.models.ReportGenerator,` 8
- `project.models.ReportType,` 9
- `project.models.SpendingHistory,` 9
- `project.models.SpendingInstance,` 9
- `project.models.SpendingInstanceFactory,`
10
- `project.models.SpendingType,` 10
- `project.models.SummaryGenerator,` 11
- `project.models.SummaryType,` 11
- `project.models.User,` 11
- `project.models.UserInformation,` 12

INDEX

A

account (project.controllers.forms.AddAccountForm attribute), 5
account (project.controllers.forms.LogSpendingForm attribute), 5
account (project.models.SpendingInstance.SpendingInstance attribute), 10
account_names (project.models.User.User attribute), 11
AccountEmailDecorator (class in project.models.Account), 7
accounts (project.models.User.User attribute), 11
AccountSavingsDecorator (class in project.models.Account), 7
add_spending_instance()
(project.models.SpendingHistory.SpendingHistory method), 9
AddAccountController (class in project.controllers.AddAccountController), 3
AddAccountForm (class in project.controllers.forms), 5
Address (class in project.models.Address), 8
address (project.models.UserInformation.UserInformation attribute), 12
amount (project.controllers.forms.LogSpendingForm attribute), 5
amount (project.models.SpendingInstance.SpendingInstance attribute), 10

B

BaseAccount (class in project.models.Account), 7
BaseAccountDecorator (class in project.models.Account), 7
BaseAddressForm (class in project.controllers.forms), 5
BaseUserForm (class in project.controllers.forms), 5

C

city (project.controllers.forms.BaseAddressForm attribute), 5
city (project.models.Address.Address attribute), 8
ConcreteAccount (class in project.models.Account), 8
CreateReportController (class in project.controllers.CreateReportController), 3

CreateReportForm (class in project.controllers.forms), 5
CSV (project.models.ReportType.ReportType attribute), 9
CSVReportGenerator (class in project.models.ReportGenerator), 8

D

date (project.controllers.forms.LogSpendingForm attribute), 5
date (project.models.SpendingInstance.SpendingInstance attribute), 10
DATE_ORIENTED (project.models.SummaryType.SummaryType attribute), 11
DateOrientedSummaryGenerator (class in project.models.SummaryGenerator), 11
decorators (project.controllers.AddAccountController.AddAccountController attribute), 3
decorators (project.controllers.CreateReportController.CreateReportController attribute), 3
decorators (project.controllers.HomeController.HomeController attribute), 3
decorators (project.controllers.SettingsController.SettingsController attribute), 4
decorators (project.controllers.SpendingController.SpendingController attribute), 5
default_filename (project.models.ReportGenerator.ReportGenerator attribute), 9
description (project.controllers.forms.LogSpendingForm attribute), 5
description (project.models.SpendingInstance.SpendingInstance attribute), 10
DINING (project.models.SpendingType.SpendingType attribute), 11
DiningSpendingInstance (class in project.models.SpendingInstance), 9
dispatch_request() (project.controllers.AddAccountController.AddAccountController method), 3
dispatch_request() (project.controllers.CreateReportController.CreateReportController method), 3
dispatch_request() (project.controllers.HomeController.HomeController method), 3
dispatch_request() (project.controllers.LoginController.LoginController

method), 4
 dispatch_request() (project.controllers.LoginController.LoginController method), 4
 dispatch_request() (project.controllers.SettingsController.SettingsController method), 4
 dispatch_request() (project.controllers.SignupController.SignupController method), 4
 dispatch_request() (project.controllers.SpendingController.SpendingController method), 5
 is_active() (project.models.User.User method), 12
 is_authenticated() (project.models.User.User method), 12
 is_authenticated() (project.models.User.User method), 12
 J
 JSON (project.models.ReportType.ReportType attribute), 9
 JSONReportGenerator (class in project.models.ReportGenerator), 8

E

email (project.controllers.forms.BaseUserForm attribute), 5
 email (project.models.User.User attribute), 11
 expenditure (project.models.Account.BaseAccount attribute), 7

F

factory_method() (project.models.SpendingInstanceFactory.SpendingInstanceFactory static method), 10
 filename (project.controllers.forms.CreateReportForm attribute), 5

G

generate_report() (project.models.ReportGenerator.CSVReportGenerator method), 8
 generate_report() (project.models.ReportGenerator.JSONReportGenerator method), 8
 generate_report() (project.models.ReportGenerator.ReportGenerator method), 9
 get_account() (project.models.User.User method), 11
 get_id() (project.models.User.User method), 12
 L
 LoginController (class in project.controllers.LoginController), 4
 LogoutController (class in project.controllers.LogoutController), 4
 LogSpendingForm (class in project.controllers.forms), 5
 M
 methods (project.controllers.AddAccountController.AddAccountController attribute), 3
 methods (project.controllers.CreateReportController.CreateReportController attribute), 3
 methods (project.controllers.LoginController.LoginController attribute), 4
 methods (project.controllers.LogoutController.LogoutController attribute), 4
 methods (project.controllers.SettingsController.SettingsController attribute), 4
 methods (project.controllers.SignupController.SignupController attribute), 4
 methods (project.controllers.SpendingController.SpendingController attribute), 5

H

HomeController (class in project.controllers.HomeController), 3
 N
 name (project.models.Account.BaseAccount attribute), 7

I

id (project.models.SpendingInstance.SpendingInstance attribute), 10
 include_description (project.controllers.forms.CreateReportForm attribute), 5
 include_email (project.controllers.forms.AddAccountForm attribute), 5
 include_savings (project.controllers.forms.AddAccountForm attribute), 5
 income (project.controllers.forms.SettingsForm attribute), 6
 income (project.controllers.forms.SignupForm attribute), 6
 income (project.models.UserInformation.UserInformation attribute), 12
 information (project.models.User.User attribute), 12
 P
 password (project.controllers.forms.BaseUserForm attribute), 5
 password (project.models.User.User attribute), 12
 project.controllers.AddAccountController (module), 3
 project.controllers.CreateReportController (module), 3
 project.controllers.forms (module), 5
 project.controllers.HomeController (module), 3
 project.controllers.LoginController (module), 4
 project.controllers.LogoutController (module), 4
 project.controllers.SettingsController (module), 4
 project.controllers.SignupController (module), 4
 project.controllers.SpendingController (module), 5
 project.models.Account (module), 7
 project.models.Address (module), 8
 project.models.ReportGenerator (module), 8
 project.models.ReportType (module), 9

project.models.SpendingHistory (module), 9
 project.models.SpendingInstance (module), 9
 project.models.SpendingInstanceFactory (module), 10
 project.models.SpendingType (module), 10
 project.models.SummaryGenerator (module), 11
 project.models.SummaryType (module), 11
 project.models.User (module), 11
 project.models.UserInformation (module), 12

R

report_generator (project.models.User.User attribute), 12
 report_type (project.controllers.forms.SettingsForm attribute), 6
 ReportGenerator (class in project.models.ReportGenerator), 9
 ReportType (class in project.models.ReportType), 9
 RETAIL (project.models.SpendingType.SpendingType attribute), 11
 RetailSpendingInstance (class in project.models.SpendingInstance), 9

S

SettingsController (class in project.controllers.SettingsController), 4
 SettingsForm (class in project.controllers.forms), 5
 SignupController (class in project.controllers.SignupController), 4
 SignupForm (class in project.controllers.forms), 6
 spending_history (project.models.User.User attribute), 12
 spending_instances (project.models.SpendingHistory.SpendingHistory attribute), 9
 spending_type (project.controllers.forms.LogSpendingForm attribute), 5
 SpendingController (class in project.controllers.SpendingController), 5
 SpendingHistory (class in project.models.SpendingHistory), 9
 SpendingInstance (class in project.models.SpendingInstance), 10
 SpendingInstanceFactory (class in project.models.SpendingInstanceFactory), 10
 SpendingType (class in project.models.SpendingType), 10
 state (project.controllers.forms.BaseAddressForm attribute), 5
 state (project.models.Address.Address attribute), 8
 street_address (project.controllers.forms.BaseAddressForm attribute), 5
 street_address (project.models.Address.Address attribute), 8
 summary_generator (project.models.User.User attribute), 12

summary_template_method() (project.models.SummaryGenerator.SummaryGenerator method), 11
 summary_type (project.controllers.forms.SettingsForm attribute), 6
 SummaryGenerator (class in project.models.SummaryGenerator), 11
 SummaryType (class in project.models.SummaryType), 11

T

TOTAL_EXPENDITURE (project.models.SummaryType.SummaryType attribute), 11
 TotalExpenditureSummaryGenerator (class in project.models.SummaryGenerator), 11

U

User (class in project.models.User), 11
 UserInformation (class in project.models.UserInformation), 12

W

withdraw() (project.models.Account.AccountEmailDecorator method), 7
 withdraw() (project.models.Account.AccountSavingsDecorator method), 7
 withdraw() (project.models.Account.BaseAccount method), 7
 withdraw() (project.models.Account.BaseAccountDecorator method), 7
 withdraw() (project.models.Account.ConcreteAccount method), 8

Z

zip (project.controllers.forms.BaseAddressForm attribute), 5
 zip (project.models.Address.Address attribute), 8