Project Title: World Cuisine App
Team Members: Kai Drumm, Myles Cork, Robert Meikle

Work Done:
Myles
- Built static pages for web app
- Setup firebase data store
- Can view individual recipes

Kai
- Implemented spoonacular api query functionality
- Can save recipes to firebase
- Adapter for pulling data from both firebase and spoonacular depending on if data already exists in firebase

Robert
- Save recipes pulled from spoonacular to firebase
- Ability to rate recipes and store ratings in firebase

Changes or Issues:
- Observer/Singleton may not be relevant because this code runs in the browser, so we never have multiple users accessing the same code flow (they access the same Firebase database).
- Added HomePage, which takes over responsibility for managing page content from the Navbar component
- QueryFacade -> RecipeManager and FirebaseUtils
- Users are only stored in Firebase, not as an object currently

Design Patterns:
- Chain of Command - various Components
- Adapter - SpoonacularAdapter
- Facade - RecipeManager
- Memento?
- MVC?

Plan for next iteration:
- Create "add a note to a recipe" functionality which uses a Decorator pattern to add on to the recipe description
- Change FirebaseUtils into a "FirebaseAdaptor" class
- Display top recipes pulled from firebase based on ratings
- View users' favorite recipes based on ratings pulled from firebase
- Create recipe functionality or change into a history tab that logs current session actions with a logger/observer pattern

Composite / Chain of Responsibility

Rating
- Timestamp: TimeDate
+ UserID: int
+ RecipeID: int
+ Value: int

Recipe
+ id: String
+ Title: String
+ Image: String

1...* — 1

UserManager
+ getLoggedInUserId(): int
+ getLoggedInUser(): Object

RatingManager
+ ratingConverter.toFirestore(Rating): Object
+ ratingConverter.fromFirestore(Object): Rating
+ searchFirebase(): array<Ratings>
+ getRating(int, int): Rating
+ addNewRating(int, int, int): void

RecipeManager
+ recipeConverter.toFirestore(Recipe): Object
+ recipeConverter.fromFirestore(Object): Recipe
+ arrayFromApiResults(array<Object>): array<Recipe>
+ saveRecipes(array<Recipe>, String): void
+ queryCuisine(String): array<Recipe>

Adapter

FirebaseAdapter
firebaseConfig: Object
+ logInWithEmailAndPassword(STring, String): void
+ registerWithEmailAndPassword(String, String, String): void
+ sendPasswordReset(String): void
+ logout(): void
+ getAuth(): Object
+ getDB(): Object

SpoonacularAdapter
+ spoonacularConfig: Object
+ cuisineSearch(String): array<Object>

<<Component>> HomePage
+ Name: String
+ render(): Component
+ openRecipe(int): void

<<Component>> SearchPage
- state.results: array<Recipe>
- state.cuisine: String
+ render( ): Component
+ handleCuisineSelect(String): void

<<Component>> SearchMenu
- onSelect: function
+ render( ): Component
-props.handleCuisineSelect(string): void

<<Component>> NavBar
+ render( ): Component

<<Component>>
render(): Component

<<Component>> SearchList
props.searchResults: array<Recipe>
openRecipe: function
+ render( ): Component

1 — 1...*

<<Component>> RecipeView
- state.user_id: int
- state.current_rating: int
- props.recipe: Recipe
- visible: boolean
- props.openRecipe: function
+ render( ): Component
+ Rate(String, int): void
+ componentDidUpdate(): void

<<Component>> TopPage
+ render( ): Component

<<Component>> SearchResult
- props.recipe: Recipe
- props.key: int
-props.openRecipe: function
+ render( ): Component

<<Component>> CreateForm
+ render( ): Component

<<Component>> FavPage
+ render( ): Component

<<Component>> TopList
+ render( ): Component

<<Component>> TopResult
+ render( ): Component

<<Component>> CreatePage
+ render( ): Component

<<Component>> FavList
+ render( ): Component

<<Component>> FavResult
+ render( ): Component

1...1
1...*
1...1
1...1
1