Code Snippets

Design Patterns:

- Factory design pattern: PlanFactory

```
public static Optional<Plan> getPlan(String name) {
    String filename = path + name + ".xml";
    Optional<Plan> optionalPlan = XMLToObject(filename);

    if(!optionalPlan.isPresent()) {
        return optionalPlan;
    }

    ClassDB db = new ClassDB();
    db.scrapeAll();
    Plan plan = optionalPlan.get();

    List<Category> categories = plan.getCategories();

    for(int i = 0; i < categories.size(); i++) {
        Category category = categories.get(i);

        List<Course> emptyCourses = category.getCourses();
        List<Course> filledCourses = new ArrayList<>();

        for(Course course : emptyCourses) {
            String courseName = course.getName();
            Course filledCourse = db.getCourse(courseName);

            if(filledCourse != null) {
                filledCourses.add(filledCourse);
            }
        }

        category.setCourses(filledCourses);
    }

    return Optional.of(plan);

}
```

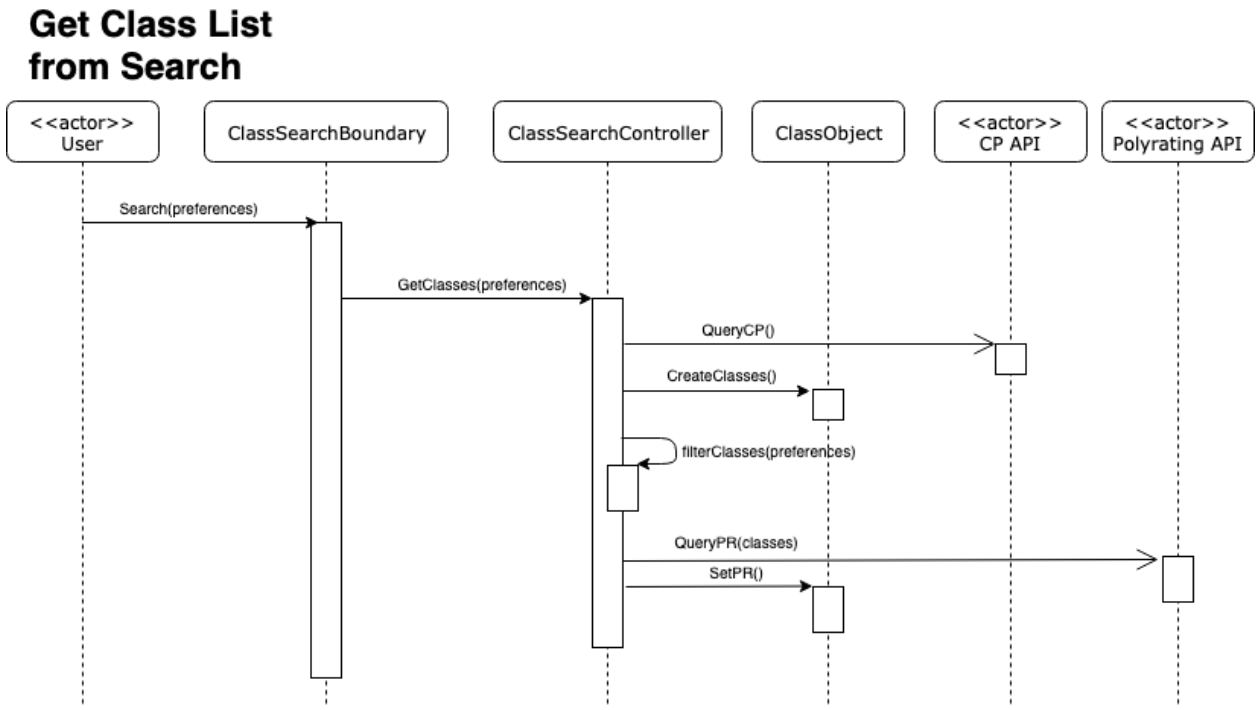
- Singleton: ClassDB

```java
public class ClassDB {
    private List<Course> courses;
    private static ClassDB instance;

    private ClassDB() {
        courses = new ArrayList<>();
    }

    public static synchronized ClassDB getInstance() {
        if(instance == null)
            instance = new ClassDB();
        return instance;
    }
```

Sequence Diagram:



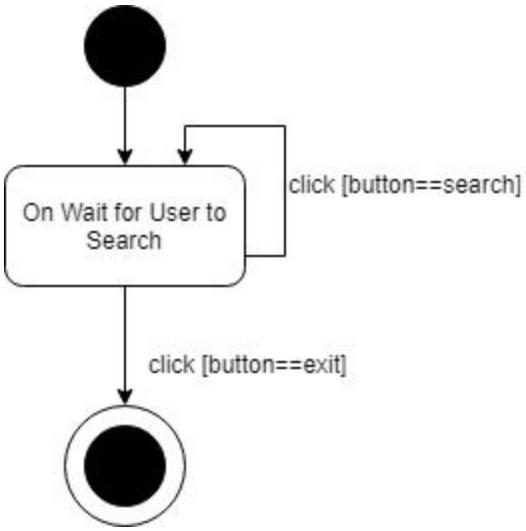
```
public static void searchCourses(String dept) {
    CoursesPane.clearItems();
    sections = FXCollections.observableArrayList();
    courses = ClassDB.filterDepartment(courses, dept);
    for(int i = 0; i < courses.size(); i++) {
        List<Section> sects = (List<Section>)courses.get(i).getSections();
        for(Section s : sects) {
            if(selectedCategory.getValue().getCourses().contains(courses.get(i))){
                sections.add(new CheckedSection(s, true));
            }else {
                sections.add(new CheckedSection(s, false));
            }
        }
    }
    CoursesPane.addItems(sections);
}
```

Activity Diagram: Edit schedule

```
private void addListeners() {
    selectedPlan.addListener((obs, newVal, oldVal)->{
        CoursesPane.clearItems();
        for(int i=0; i < catPane.getChildren().size(); i++) {
            catPane.getChildren().remove(0);
        }
        catPane = CategoryPane.getCategoryPane(selectedPlan);
        for(int i=0; i < pane2.getChildren().size(); i++) {
            pane2.getChildren().remove(0);
        }
        pane2.add(catPane, 0, 0);
    });
    selectedCategory.addListener((obs, newVal, oldVal)->{
        CoursesPane.clearItems();
        sections = FXCollections.observableArrayList();
        for(int i = 0; i < courses.size(); i++) {
            ArrayList<Section> sects = (ArrayList<Section>)courses.get(i).getSections();
            for(Section s : sects) {
                if(selectedCategory.getValue().getCourses().contains(courses.get(i))){
                    sections.add(new CheckedSection(s, true));
                }else {
                    sections.add(new CheckedSection(s, false));
                }
            }
        }
        CoursesPane.addItems(sections);
    });
    sections.addListener(new ListChangeListener<CheckedSection>() {
```

```
@Override
public void onChanged(ListChangeListener.Change<? extends CheckedSection> col) {
    while (col.next()) {
        if (col.wasUpdated()) {
            int param = col.getFrom();
            if(sections.get(param).getChecked()) {
                for(Course c : courses) {
                    if(c.getName() == sections.get(param).getCourseName() &&
                            !selectedCategory.getValue().getCourses().contains(c)) {
                        selectedCategory.getValue().addCourse(c);
                    }
                }
            } else {
                for(Course c : courses) {
                    if(c.getName() == sections.get(param).getCourseName() &&
                            !selectedCategory.getValue().getCourses().contains(c)) {
                        selectedCategory.getValue().removeCourse(c);
                    }
                }
            }
            CoursesPane.clearItems();
            sections = FXCollections.observableArrayList();
            for(int i = 0; i < courses.size(); i++) {
                ArrayList<Section> sects = (ArrayList<Section>)courses.get(i).getSections();
                for(Section s : sects) {
                    if(selectedCategory.getValue().getCourses().contains(courses.get(i))){
                        sections.add(new CheckedSection(s, true));
                    }else {
                        sections.add(new CheckedSection(s, false));
                    }
                }
            }
            CoursesPane.addItems(sections);
        }
    }
}
```

State Diagram Snippet:



```
32      public static GridPane getCoursesPane() {
33          coursePane.setAlignment(Pos.TOP_CENTER);
34          coursePane.getStyleClass().add("borders");
35          coursePane.getStyleClass().add("outside");
36
37          TextField textField = new TextField();
38          textField.setPromptText("Search here!");
39          textField.requestFocus();
40          textField.getStyleClass().add("search-bar");
41          textField.getStyleClass().add("borders");
42          coursePane.add(textField, 0, 0);
43          Button searchButton = new Button("Search");
44          searchButton.setOnMouseClicked(new EventHandler<MouseEvent>() {
45              @Override public void handle(MouseEvent e) {
46                  Gui.searchCourses(textField.textProperty().getValue());
47              }
48          });
49          coursePane.add(searchButton, 1, 0);
```