iDrive - ΝΝΑ-06-04 ΑΝΑΠΤΥΞΗ ΕΦΑΡΜΟΓΩΝ ΚΙΝΗΤΩΝ ΣΥΣΚΕΥΩΝ ΕΡΓΑΣΙΑ

Εργασία Android από:

4442 Βασίλης Ζερβός

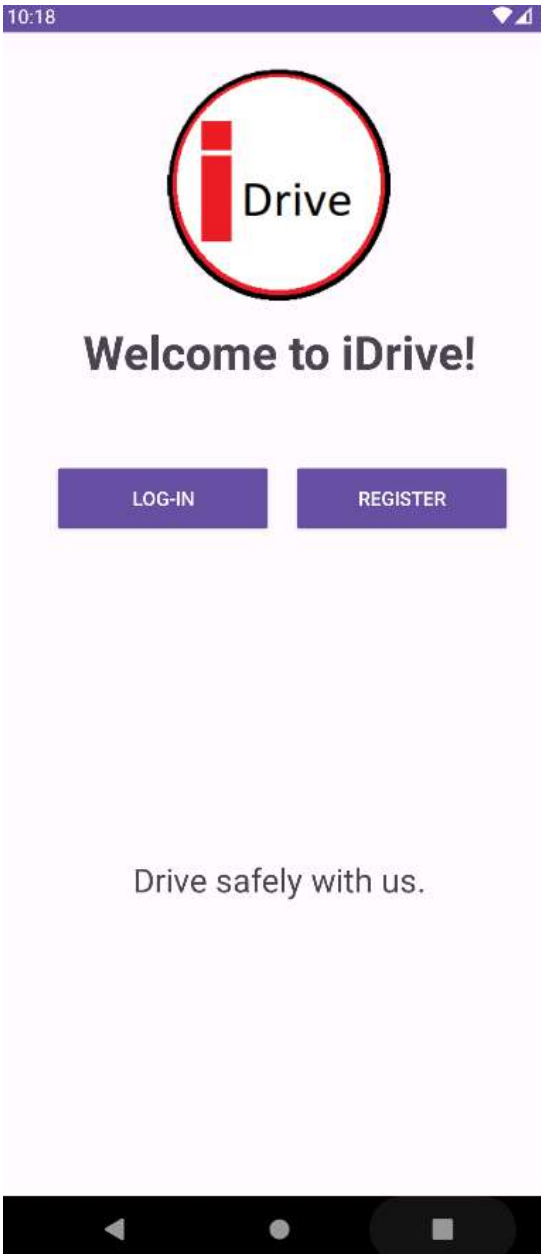
4179 Βέργος Γεώργιος

100069 Κωνσταντίνος Γεωργίου

Η εφαρμογή iDrive είναι ένα βοηθητικό εργαλείο για μια σχολή οδηγών, με τον σκοπό την διευκόλυνση κανονισμού μαθημάτων μεταξύ μαθητών και δάσκαλο.

1. Αρχική σελίδα

   Η εφαρμογή ανοίγει με μία activity StartPageActivity η οποία προσφέρει δύο επιλογές, LOG-IN & REGISTER, και η κάθε μία σε πηγαίνει στο κατάλληλο fragment της AuthActivity όπου χειρίζεται η ταυτοποίηση.

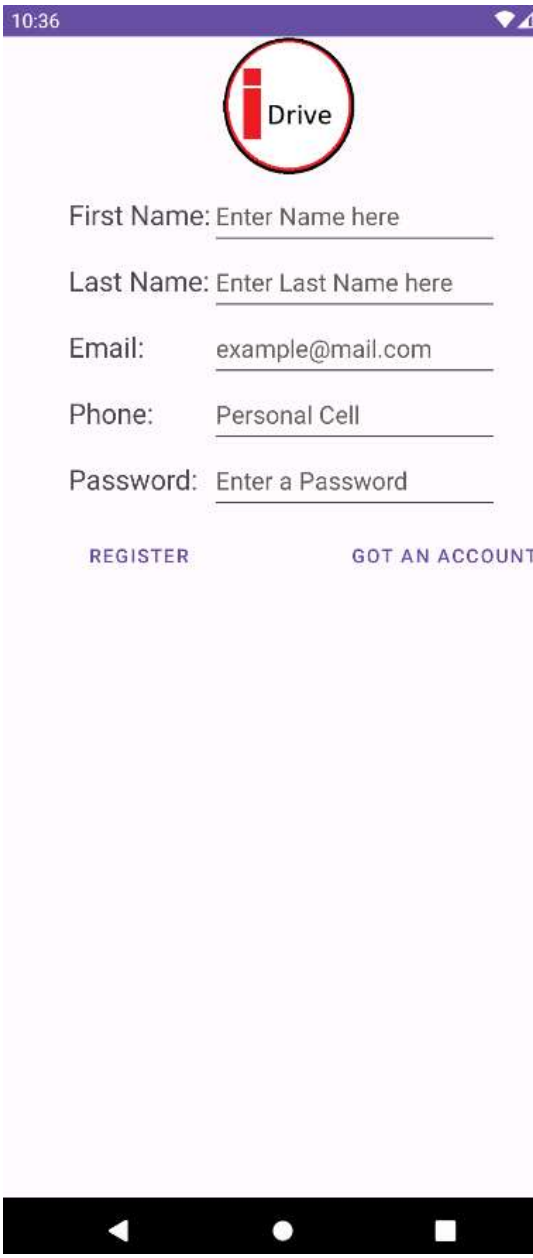


2. AuthActivity / LoginFragment

   Για τους χρήστες που έχουν λογαριασμό, μπορούν να συνδεθούν με το email και το password που δήλωσαν.

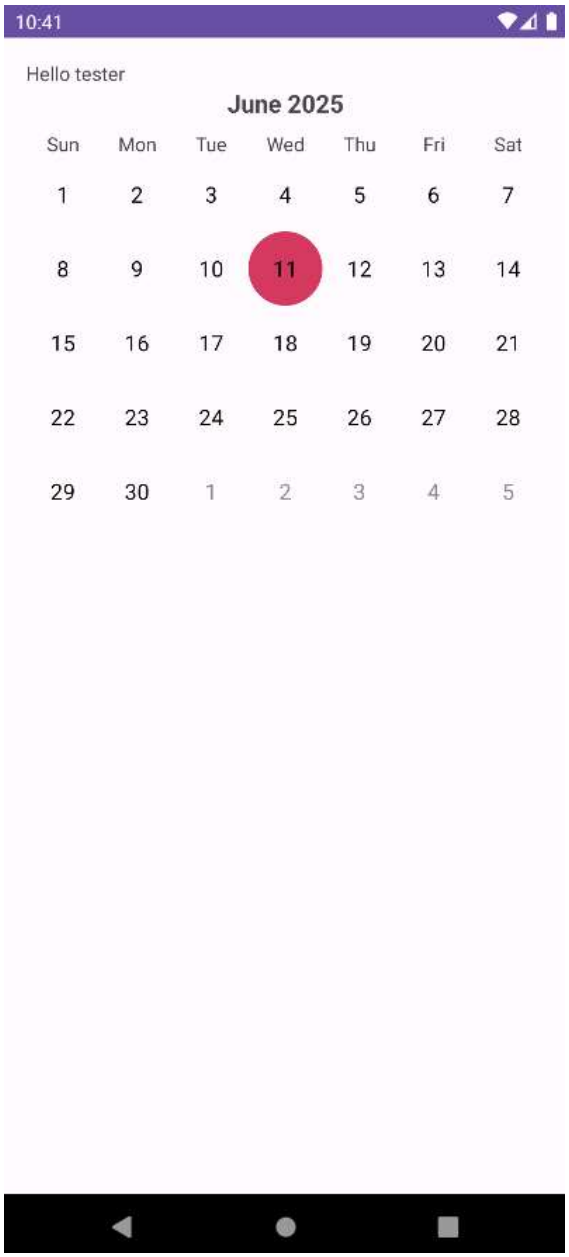
3. AuthActivity / RegisterFragment

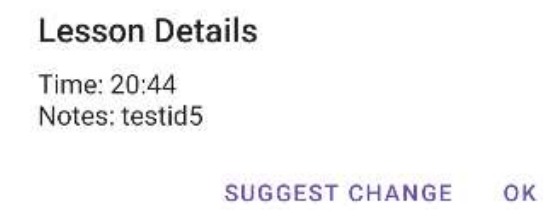
   Για τους χρήστες που δεν έχουν λογαριασμό, πρέπει να σημπληρώσουν το όνομα και επώνυμό τους, ένα email, ένα τηλέφωνο και τον κωδικό τους.

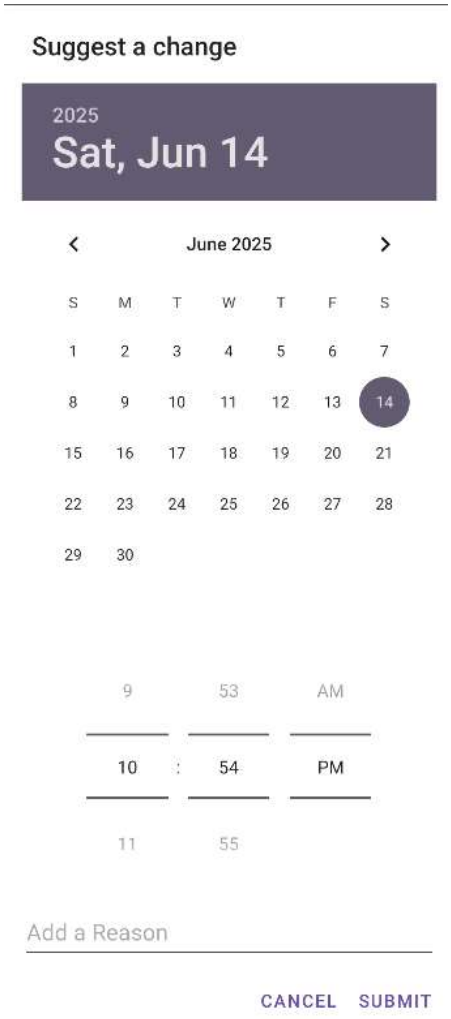


4. HomePageActivity / MemberHomeFragment

Ως μαθητής, μπορείς να δεις ένα χαιρετισμό και το ημερολόγιο, με τις ημέρες που έχεις μάθημα τονισμένες με κόκκινο χρώμα. Υπάρχει λειτουργία για προτάσεις αλλαγής μέρας ή ώρας όταν πατήσεις μία τονισμένη μέρα.

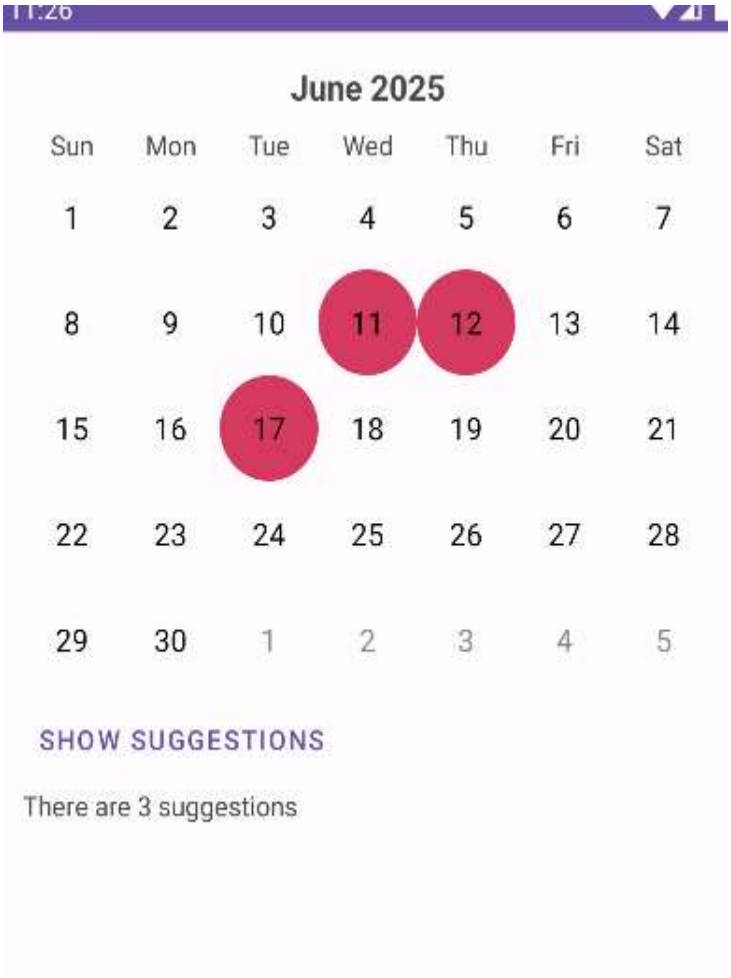


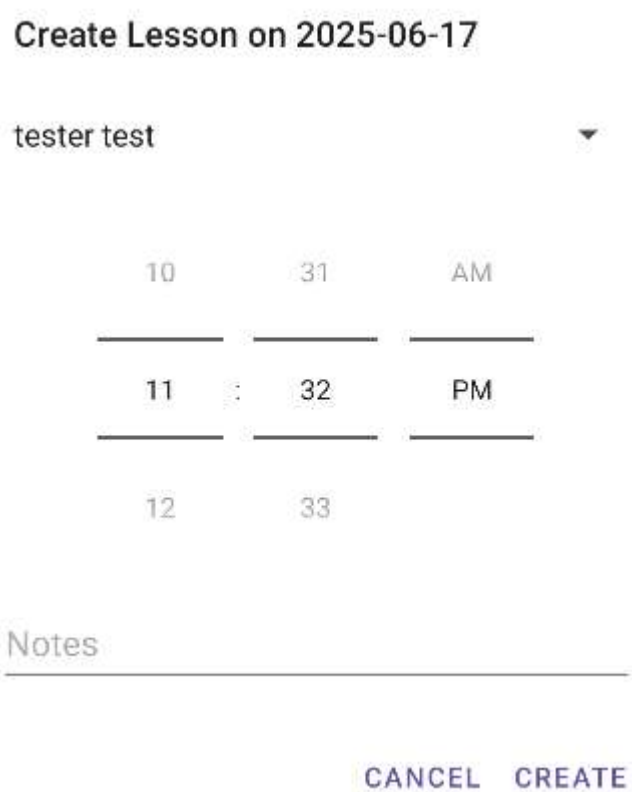


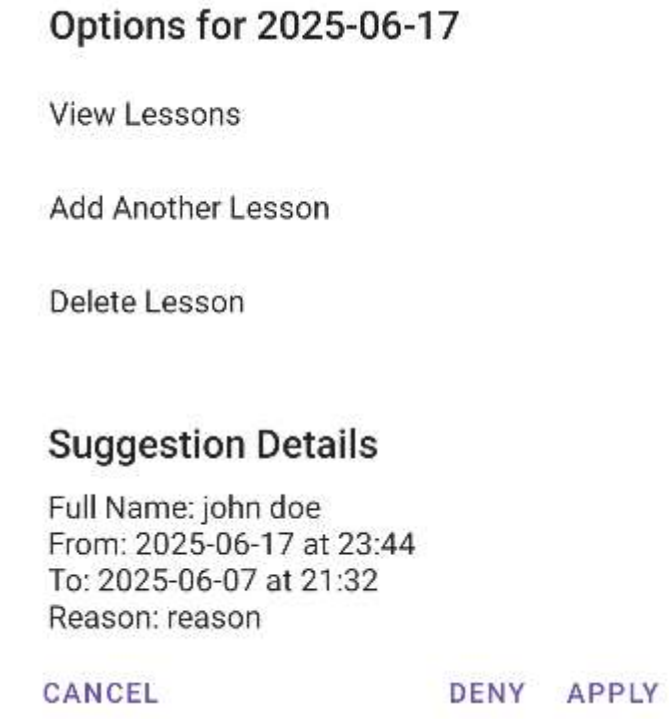


5. HomePageActivity / AdminHomeFragment

Ο διαχειρηστής έχει τις εξήσου επιλογές; να θέσει νέο μάθημα στην επιλεγμένη μέρα, να διαγράψει ένα μάθημα και να χειριστεί προτάσεις των μαθητών.







Συνοπτική περιγραφή:

1) Activites:
    a) StartPageActivity
    b) AuthActivity
    c) HomePageActivity
2) Data:
    a) Db:
        i) FirebaseHandler
    b) Models:
        i) Lesson
        ii) User
        iii) Suggestion
3) Fragments:
    a) AdminHomeFragment (HomePageActivity)
    b) MemberHomeFragment (HomePageActivity)
    c) LoginFragment (AuthActivity)
    d) RegisterFragment (AuthActivity)

Το ημερολόγιο είναι από τον kizitonwose, που προσφέρει ένα πιο καθαρό και προσαρμόσιμο UI από το ημερολόγιο widget της android.

https://github.com/kizitonwose/Calendar

Για βάση δεδομένων cloud χρησιμοποιήσαμε το Firebase

https://console.firebase.google.com/u/0/

** Η εργασία και η τεχνική αναφορά έγινε μόνο από εμένα, Βασίλη Ζερβό 4442

Ανάλυση του κώδικα:

StartPage Activity:

```java
public class StartPageActivity extends AppCompatActivity {
    2 usages
    Button loginBtn, registerBtn;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_start);
        loginBtn = findViewById(R.id.btnLogin);
        registerBtn = findViewById(R.id.btnRegister);

        loginBtn.setOnClickListener( View view -> {
            Intent intent = new Intent( packageContext: StartPageActivity.this, AuthActivity.class);
            intent.putExtra( name: "auth_type",  value: "login");
            startActivity(intent);
        });

        registerBtn.setOnClickListener( View view -> {
            Intent intent = new Intent( packageContext: StartPageActivity.this, AuthActivity.class);
            intent.putExtra( name: "auth_type",  value: "register");
            startActivity(intent);
        });
    }
}
```

Η αρχή της εφαρμογής, αρχικοποιούμε τα δύο κουμπιά loginBtn, registerBtn από την αρχή και έπειτα, το κάθε κουμπί μεταφέρει τον χρήστη στο επόμενο activity αξιοποιώντας το intent, με αποτέλεσμα το auth_type να καθοδηγεί τον χρήστη στο κατάλληλο fragment.

Auth Activity:

```java
public class AuthActivity extends AppCompatActivity {
    @Override
    public void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_auth);

        if (savedInstanceState == null) {
            String authType = getIntent().getStringExtra( name: "auth_type");
            Fragment fragment = "register".equals(authType) ? new RegisterFragment() : new LoginFragment();
            getSupportFragmentManager().beginTransaction()
                    .replace(R.id.fcvAuth, fragment)
                    .commit();
        }
    }

    2 usages
    public void loadFragment(Fragment fragment) {
        getSupportFragmentManager() FragmentManager
                .beginTransaction() FragmentTransaction
                .replace(R.id.fcvAuth, fragment)
                .commit();
    }
}
```

Σε αυτό το activity χειριζόμαστε την ταυτοποίηση του χρήστη. Αρχικά, με το intent που αποκτήσαμε πατώντας ένα από τα δύο κουμπιά στην StartPage, δημιουργούμε και εμφανίζουμε το κατάλληλο fragment μέσα στο fcvAuth; FragmentContainerView. Άμα χρειαστεί να αλλάξουμε από το ένα fragment στο άλλο (π.χ. ο χρήστης πάτησε login αλλά δεν έχει λογαριασμό) τότε χρησιμοποιούμε την βοηθητική μέθοδο loadFragment(Fragment fragment), ως παράμετρο το fragment που θέλουμε να δημιουργήσουμε.

HomePage Activity:

```java
public class HomePageActivity extends AppCompatActivity {
    @Override
    public void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);

        if (savedInstanceState == null) {
            boolean isAdmin = getIntent().getBooleanExtra( name: "is_admin",  defaultValue: false);
            Fragment fragment = isAdmin ? new AdminHomeFragment() : new MemberHomeFragment();
            getSupportFragmentManager().beginTransaction()
                    .replace(R.id.fcvHome, fragment)
                    .commit();
        }
    }
}
```

Το τελικό activity της εφαρμογής αλλά και το πιο σημαντικό. Με την δημιουργία του, παρατηρούμε άμα ο χρήστης είναι διαχειριστής ή μαθητής μέσω του intent : booleanExtra is_admin, το οποίο αρχικοποιούμε ως false και κατασκευάζουμε το ανάλογο fragment μέσα στο fcvHome; FragmentContainerView

Register Fragment:

Αν το auth_type γυρίσει με value “register”, τότε συνεχίζουμε από το StartPage Activity στο register fragment της Auth Activity. Για να δημιουργήσεις λογαριασμό χρειάζεται να συμπληρώσεις όλα τα editText fields όπως σου εξηγούν τα hints και μετά από έναν απλό έλεγχο ορθότητας, φτιάχνει έναν νέο user με τη βοηθητική μέθοδο registerUser και σε μεταφέρει στο homepage ως μαθητή με την goToHomepage.

```
btnRegister.setOnClickListener( View v -> {
    String firstName = etFirstName.getText().toString().trim();
    String lastName = etLastName.getText().toString().trim();
    String email = etEmail.getText().toString().trim();
    String password = etPassword.getText().toString().trim();
    String phone = etPhone.getText().toString().trim();

    if (firstName.isEmpty() || lastName.isEmpty() || email.isEmpty() ||
            phone.isEmpty() || password.isEmpty()) {
        Toast.makeText(getContext(), text: "Please fill all fields", Toast.LENGTH_SHORT).show();
        return;
    }
    if (!Patterns.EMAIL_ADDRESS.matcher(email).matches()) {
        etEmail.setError("Invalid email");
        etEmail.requestFocus();
        return;
    }
    if (!Patterns.PHONE.matcher(phone).matches()) {
        etPhone.setError("Invalid Phone");
        etPhone.requestFocus();
        return;
    }
    if (password.length() < 6) {
        etPassword.setError("Password must be at least 6 characters");
        etPassword.requestFocus();
        return;
    }

    registerUser(firstName, lastName, email, phone, password);
```

```java
private void registerUser(String firstName, String lastName, String email, String phone, String password) {
    auth.createUserWithEmailAndPassword(email, password).addOnCompleteListener( Task<AuthResult> task -> {
        if (task.isSuccessful()) {
            FirebaseUser firebaseUser = auth.getCurrentUser();
            if (firebaseUser == null) return;
            String uid = firebaseUser.getUid();
            User newUser = new User(firstName, lastName, phone, email, password);
            newUser.setUid(uid);
            handler.addUser(newUser, firebaseUser.getUid());
            goToHomepage(newUser);
        } else {
            Toast.makeText(getContext(), text: "Registration failed: " + task.getException().getMessage(), Toast.LENGTH_SHORT).show();
        }
    });
}

1 usage
private void goToHomepage(User user) {
    Intent intent = new Intent(getActivity(), HomePageActivity.class);
    intent.putExtra( name: "is_admin", user.getIsAdmin());
    startActivity(intent);
    getActivity().finish();
}
```

Login Fragment:

Αν το auth_type γυρίσει με value “login”, τότε συνεχίζουμε από το StartPage Activity στο login fragment της Auth Activity. Αφού ο χρήστης έχει ήδη λογαριασμό και επειδή για βάση δεδομένων χρησιμοποιούμε την Firebase, μπορούμε να καλέσουμε μία μέθοδο της FirebaseAuth; ένα official plugin της Firebase. Η μέθοδος signInWithEmailAndPassword μας αρκεί για την ταυτοποίηση του χρήστη και τέλος, μας μεταφέρει στην HomePage Activity με το Intent extra “is_admin”, μετά από έλεγχο του χρήστη για το άμα είναι admin ή όχι.

```java
private void tryLogin() {
    String email = etEmail.getText().toString().trim();
    String password = etPassword.getText().toString().trim();

    if (email.isEmpty() || password.isEmpty()) {
        Toast.makeText(getContext(), text: "Please fill in all fields", Toast.LENGTH_SHORT).show();
        return;
    }

    authenticator.signInWithEmailAndPassword(email, password)
            .addOnCompleteListener( Task<AuthResult> task -> {
                if (task.isSuccessful()) {
                    FirebaseUser user = authenticator.getCurrentUser();
                    if (user == null) {
                        Toast.makeText(getContext(), text: "Authentication error. Please try again.", Toast.LENGTH_SHORT).show();
                        return;
                    }
                    FirebaseFirestore.getInstance() FirebaseFirestore
                            .collection( collectionPath: "users") CollectionReference
                            .document(user.getUid()) DocumentReference
                            .get() Task<DocumentSnapshot>
                            .addOnSuccessListener( DocumentSnapshot documentSnapshot -> {
                                if (documentSnapshot.exists()) {
                                    boolean isAdmin = documentSnapshot.getBoolean( field: "isAdmin") != null
                                            && documentSnapshot.getBoolean( field: "isAdmin");
                                    Intent intent = new Intent(getActivity(), HomePageActivity.class);
                                    intent.putExtra( name: "is_admin", isAdmin);
                                    startActivity(intent);
                                    requireActivity().finish();
                                } else {
                                    Toast.makeText(getContext(), text: "User data not found. Please register first.", Toast.LENGTH_SHORT).show();
                                }
                            });
                } else {
                    Toast.makeText(getContext(), text: "Login failed: " + task.getException().getMessage(), Toast.LENGTH_LONG).show();
                }
            });
```

MemberHome Fragment:

Είναι υπεύθυνο για την κύρια σελίδα των μελών, η οποία εμφανίζει ένα διαδραστικό ημερολόγιο μαθημάτων και προσφέρει τη δυνατότητα πρότασης αλλαγής σε μάθημα.

CalendarView

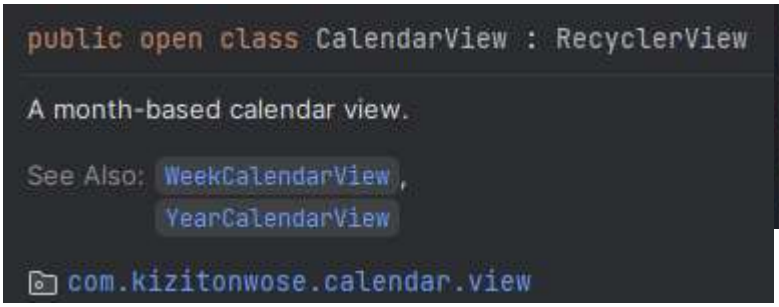
Με χρήση της βιβλιοθήκης kizitonwose/calendar, γίνεται εμφάνιση ενός ημερολογίου που φορτώνει τα μαθήματα του χρήστη από τη Firebase, και τα εμφανίζει με τονισμό στις ημερομηνίες των μαθημάτων. Το ημερολόγιο είναι προσαρμοσμένο ώστε να δείχνει μόνο τα μαθήματα του logged-in user.

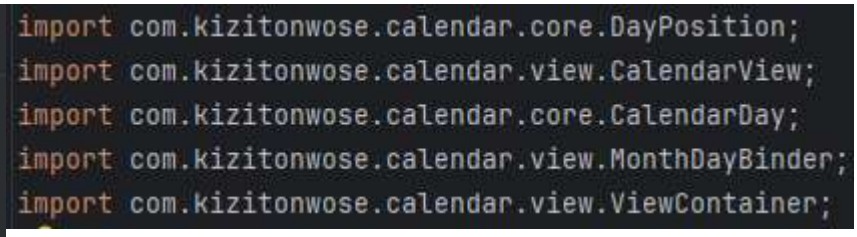
showLessonDialog

Με πάτημα σε ημερομηνία που περιέχει μάθημα, εμφανίζεται ένα dialog που δείχνει την ώρα και τις σημειώσεις του μαθήματος. Αν υπάρχουν περισσότερα από ένα, εμφανίζονται ως λίστα επιλογών.

showSuggestDialog

Μέσα από το ίδιο παράθυρο διαλόγου, ο χρήστης μπορεί να πατήσει "Suggest Change" και να ανοίξει ένα νέο dialog όπου επιλέγει νέα ημερομηνία, ώρα και προσθέτει τον λόγο για την αλλαγή. Πριν εμφανιστεί το dialog, ελέγχεται στη Firebase αν έχει ξανακάνει suggestion στην ίδια ημερομηνία. Αν υπάρχει, μπλοκάρεται η δημιουργία δεύτερης πρότασης. Η νέα πρόταση αποθηκεύεται με αναφορά στο μάθημα (lessonId), την προηγούμενη (oldDate, oldTime) και την προτεινόμενη ημερομηνία/ώρα (newDate, newTime).





```java
private void showLessonDialog(Lesson lesson) {
    String message = "Time: " + lesson.getTime() + "\nNotes: " + lesson.getNotes();

    new AlertDialog.Builder(requireContext())
            .setTitle("Lesson Details")
            .setMessage(message)
            .setPositiveButton( text: "OK",  listener: null)
            .setNegativeButton( text: "Suggest Change", ( DialogInterface dialog,  int which) -> {
                firebaseHandler.checkSimilarSuggestions(lesson.getUserId(), lesson.getDate(),  Boolean isUnique -> {
                    if (isUnique) {
                        showSuggestDialog(lesson);
                    } else {
                        Toast.makeText(getContext(),  text: "You've already made a suggestion on this day.", Toast.LENGTH_SHORT).show()
                    }
                },  Exception e -> Toast.makeText(getContext(),  text: "Error on multiple suggestions.", Toast.LENGTH_SHORT).show());
            })
            .show();
}
```

```java
private void showSuggestDialog(Lesson lesson) {
    View dialogView = LayoutInflater.from(requireContext())
            .inflate(R.layout.dialog_suggestion,  root: null);
    DatePicker datePicker = dialogView.findViewById(R.id.datePicker);
    TimePicker timePicker = dialogView.findViewById(R.id.timePicker);
    EditText etNotes = dialogView.findViewById(R.id.etNotes);

    new AlertDialog.Builder(requireContext())
            .setTitle("Suggest a change")
            .setView(dialogView)
            .setPositiveButton( text: "Submit", ( DialogInterface dialog,  int which) -> {
                int year = datePicker.getYear();
                int month = datePicker.getMonth() + 1;
                int day = datePicker.getDayOfMonth();
                int hour = timePicker.getHour();
                int minute = timePicker.getMinute();
                String notes = etNotes.getText().toString().trim();
                LocalDate newDate = LocalDate.of(year, month, day);
                LocalTime newTime = LocalTime.of(hour, minute);

                Suggestion suggestion = new Suggestion(
                        lesson.getUserId(),
                        lesson.getId(),
                        lesson.getDate(),
                        lesson.getTime(),
                        newDate.toString(),
                        newTime.toString(),
                        notes
                );

                firebaseHandler.addSuggestion(suggestion);
                Toast.makeText(requireContext(),  text: "Suggestion made", Toast.LENGTH_SHORT).show();
            })
            .setNegativeButton( text: "Cancel",  listener: null)
            .show();
}
```

AdminHome Fragment:

Είναι η σελίδα του διαχειριστή όπου μπορεί να κάνει αλλαγές στις ημερομηνίες όπου υπάρχουν μαθήματα (pickLessonDialog -> showLessonDialog) ή να δημιουργήσει καινούρια

για έναν μαθητή που θα επιλέξει, στην ημερομηνία που θα πατήσει (showCreateLessonDialog) ή και να διαγράψει μάθημα (showLessonDialog -> removeLesson)

Επίσης, ο διαχειριστής μπορεί να δει όλες τις προτάσεις (pickSuggestion) που κάνουν οι μαθητές, με την επιλογή να τις εφαρμόσει ή να τις απορρίψει (showSuggestionDialog).

```
private void showLessonDialog(Lesson lesson) {
    firebaseHandler.getAllMembers( Map<String, String> uidToNameMap -> {
        String fullName = uidToNameMap.getOrDefault(lesson.getUserId(),  defaultValue: "No User Found");
        String message = "User: " + fullName + "\nTime: " + lesson.getTime() + "\nNotes: " + lesson.getNotes();

        new AlertDialog.Builder(requireContext())
                .setTitle("Lesson Details")
                .setMessage(message)
                .setPositiveButton( text: "OK",  listener: null)
                .setNegativeButton( text: "REMOVE", ( DialogInterface dialog,  int which) -> {
                    firebaseHandler.deleteLesson(lesson.getId(),  Void aVoid -> {
                        List<Lesson> list = lessonByDate.get(LocalDate.parse(lesson.getDate()));
                        if (list != null) {
                            list.remove(lesson);
                            if (list.isEmpty()) {
                                lessonByDate.remove(LocalDate.parse(lesson.getDate()));
                            }
                        }
                        calendarView.notifyDateChanged(LocalDate.parse(lesson.getDate()));
                        Toast.makeText(getContext(),  text: "Lesson removed.", Toast.LENGTH_SHORT).show();
                    },  Exception e -> {
                        Toast.makeText(getContext(),  text: "Error on lesson removal", Toast.LENGTH_SHORT).show();
                    });
                })
                .show();
    },  Exception e -> Toast.makeText(getContext(),  text: "Failed loading Users", Toast.LENGTH_SHORT).show());
}
```

```
private void showCreateLessonDialog(LocalDate pickedDate) {
```

```
private void removeLesson(List<Lesson> lessons) {
    String[] items = new String[lessons.size()];
    for (int i = 0; i < lessons.size(); i++) {
        Lesson l = lessons.get(i);
        String name = uidToNameMap.getOrDefault(l.getUserId(),  defaultValue: "No User Found");
        items[i] = name + " at " + l.getTime();
    }

    new AlertDialog.Builder(requireContext())
            .setTitle("Which lesson to delete?")
            .setItems(items, ( DialogInterface dialog,  int which) -> {
                Lesson selectedLesson = lessons.get(which);
                LocalDate date = LocalDate.parse(selectedLesson.getDate());
                firebaseHandler.deleteLesson(selectedLesson.getId(),  Void aVoid -> {
                    List<Lesson> dateLessons = lessonByDate.get(date);
                    if (dateLessons != null) {
                        dateLessons.remove(selectedLesson);
                        if (dateLessons.isEmpty()) {
                            lessonByDate.remove(date);
                        }
                    }
                    calendarView.notifyDateChanged(date);
```

```java
private void showSuggestionDialog(Suggestion suggestion) {
    String fullName = uidToNameMap.getOrDefault(suggestion.getUserId(), defaultValue: "Unknown");

    String message = "Full Name: " + fullName +
            "\nFrom: " + suggestion.getOldDate() + " at " + suggestion.getOldTime() +
            "\nTo: " + suggestion.getNewDate() + " at " + suggestion.getNewTime() +
            "\nReason: " + suggestion.getReason();

    new AlertDialog.Builder(requireContext())
            .setTitle("Suggestion Details")
            .setMessage(message)
            .setPositiveButton( text: "Apply", ( DialogInterface dialog, int which) -> {
                firebaseHandler.updateLesson(suggestion.getLessonId(), suggestion, Void aVoid -> {
                    refreshSuggestionsCount();
                    LocalDate oldDate = LocalDate.parse(suggestion.getOldDate());
                    LocalDate newDate = LocalDate.parse(suggestion.getNewDate());
                    List<Lesson> oldList = lessonByDate.get(oldDate);
                    if (oldList != null) {
                        oldList.removeIf( Lesson lesson -> lesson.getId().equals(suggestion.getLessonId()));
                        if (oldList.isEmpty()) lessonByDate.remove(oldDate);
                    }
                    Lesson updatedLesson = new Lesson(
                            suggestion.getLessonId(),
                            suggestion.getUserId(),
                            suggestion.getNewDate(),
                            suggestion.getNewTime(),
                            suggestion.getReason()
                    );
                    updatedLesson.setId(suggestion.getLessonId());
                    lessonByDate.computeIfAbsent(newDate, LocalDate k -> new ArrayList<>()).add(updatedLesson);

                    calendarView.notifyDateChanged(oldDate);
                    calendarView.notifyDateChanged(newDate);
                    Toast.makeText(getContext(), text: "Suggestion applied", Toast.LENGTH_SHORT).show();
                }, Exception e -> Toast.makeText(getContext(), text: "Failed to add lesson", Toast.LENGTH_SHORT).show());
            })
            .setNegativeButton( text: "Deny", ( DialogInterface dialog, int which) -> {
                firebaseHandler.deleteSuggestion(suggestion.getId(), Void aVoid -> {
                    refreshSuggestionsCount();
```

```java
private void pickSuggestion(List<Suggestion> suggestions) {
    if (suggestions.isEmpty()) {
        Toast.makeText(getContext(), text: "No suggestions available", Toast.LENGTH_SHORT).show();
        return;
    }
    String[] suggestionItems = suggestions.stream() Stream<Suggestion>
            .map( Suggestion s -> {
                String name = uidToNameMap.getOrDefault(s.getUserId(), defaultValue: "Unknown");
                return name + " (" + s.getOldDate() + ")";
            }) Stream<String>
            .toArray(String[]::new);
    new AlertDialog.Builder(requireContext())
            .setTitle("Suggestions")
            .setItems(suggestionItems, ( DialogInterface dialog, int which) -> {
                showSuggestionDialog(suggestions.get(which));
            })
            .show();
}
```

FirebaseHandler and data:

Η class που είναι υπεύθυνη για την διασύνδεση μεταξύ της εφαρμογής με την βάση δεδομένων Firebase και την αποθήκευση στοιχείων. Χρησιμοποιείται από τα AuthActivity fragments για ταυτοποίηση, στην HomePage για να σώζει και να φορτώνει δεδομένα στο ημερολόγιο. Έχουμε 3 μοντέλα πίνακες, τα User, Lesson, Suggestion και όλη η λογική της βάσης γίνεται στην Handler class. Η Firebase απαιτεί getters και setters για κάθε μοντέλο, έναν άδειο κατασκευαστή καθώς και έναν με όλα τα στοιχεία, παρέχει ένα μοναδικό identifier σε όλα τα document του, το οποίο κάνουμε pair με το object μέσω του handler και μεθόδων στα models. Για καλύτερο error handling, χρησιμοποιούμε onSuccessListener και onFailureListener, περισσότερο στις μεθόδους που πρέπει να κάνουμε και refresh κάποια UI elements τα οποία μπορεί να μην δείχνουν σωστά δεδομένα λόγο της async λειτουργίας της Firebase.

```
public void getLessonList(
        BiConsumer<List<Lesson>, Map<String, String>> onSuccess,
        Consumer<Exception> onFailure) {
    db.collection(LESSONS).get()
            .addOnSuccessListener( QuerySnapshot querySnapshot -> {
                List<Lesson> lessons = new ArrayList<>();
                Set<String> uids = new HashSet<>();
                for (DocumentSnapshot doc : querySnapshot.getDocuments()) {
                    Lesson lesson = doc.toObject(Lesson.class);
                    if (lesson != null) {
                        lessons.add(lesson);
                        uids.add(lesson.getUserId());
                    }
                }
                db.collection(USERS).whereIn(FieldPath.documentId(), new ArrayList<>(uids)) Query
                        .get() Task<QuerySnapshot>
                        .addOnSuccessListener( QuerySnapshot userSnapshot -> {
                            Map<String, String> uidToFullName = new HashMap<>();
                            for (DocumentSnapshot doc : userSnapshot.getDocuments()) {
                                String fullName = doc.getString( field: "firstName") + " " + doc.getString( field: "lastName");
                                uidToFullName.put(doc.getId(), fullName);
                            }
                            onSuccess.accept(lessons, uidToFullName);
                        })
                        .addOnFailureListener(onFailure::accept);
            })
            .addOnFailureListener(onFailure::accept);
}
```

```java
public void addLesson(Lesson lesson, OnSuccessListener<Void> onSuccess, OnFailureListener onFailure) {
    String docId = db.collection(LESSONS).document().getId();
    lesson.setId(docId);
    db.collection(LESSONS).document(docId).set(lesson)
            .addOnSuccessListener(onSuccess)
            .addOnFailureListener(onFailure);
}
```

```java
public void addUser(User user, String uid) { db.collection(USERS).document(uid).set(user); }

1 usage
public void getUser(String uid,
                    OnSuccessListener<DocumentSnapshot> successListener,
                    OnFailureListener failureListener) {
    db.collection(USERS) CollectionReference
            .document(uid) DocumentReference
            .get() Task<DocumentSnapshot>
            .addOnSuccessListener(successListener)
            .addOnFailureListener(failureListener);
}

1 usage
public void getCurrentUser(OnSuccessListener<DocumentSnapshot> successListener,
                           OnFailureListener failureListener) {
    FirebaseUser currentUser = FirebaseAuth.getInstance().getCurrentUser();
    if (currentUser != null) {
        getUser(currentUser.getUid(), successListener, failureListener);
    } else {
        failureListener.onFailure(new Exception("No current user exists"));
    }
}
```

```java
public void addSuggestion(Suggestion suggestion) {
    String docId = db.collection(SUGGESTIONS).document().getId();
    suggestion.setId(docId);
    db.collection(SUGGESTIONS).document(docId).set(suggestion);
}

2 usages
public void getSuggestions(OnSuccessListener<QuerySnapshot> successListener,
                           OnFailureListener failureListener) {
    db.collection(SUGGESTIONS) CollectionReference
            .get() Task<QuerySnapshot>
            .addOnSuccessListener(successListener)
            .addOnFailureListener(failureListener);
}

1 usage
public void checkSimilarSuggestions(String uid, String date, OnSuccessListener<Boolean> onSuccess, OnFailureListener onFailure) {
    db.collection(SUGGESTIONS) CollectionReference
            .whereEqualTo( field: "userId",uid) Query
            .whereEqualTo( field: "oldDate", date)
            .get() Task<QuerySnapshot>
            .addOnSuccessListener( QuerySnapshot querySnapshot -> onSuccess.onSuccess(querySnapshot.isEmpty()))
            .addOnFailureListener(onFailure);
}
```

Λειτουργίες που μπορούν να υλοποιηθούν για μία σταθερή και commercial release version:

- updateSuggestion(Suggestion suggestion) μέθοδος που θα μπορούσε να χρησιμοποιηθεί όταν ένας χρήστης κάνει πρόταση αλλαγής στην ίδια ημερομηνία που έκανε πρίν, για να αλλάζει την παλιά του πρόταση.
- goToTests() μέθοδο και κουμπί στην homepage => member fragment για να μπορεί ο μαθητής να κάνει online test για προετοιμασία στην εξέταση σημάτων, αλλά και να διαβάσει τις εκδώσεις Κ.Ο.Κ. ανάλογα με το δίπλωμα που θέλει.
- changeTranslationBtn κουμπί για αλλαγή γλώσσας στην StartPage και ένα Options κουμπί στην HomePage που θα έχει αυτήν την επιλογή.
- changeThemeBtn κουμπί που να αλλάζει την εμφάνιση της εφαρμογής ανάλογα την διάθεση του χρήστη. Τώρα, αλλάζει μόνο ανάλογα την επιλογή από το σύστημα.
- (Optional) contactPage activity ή dialog στα options που θα έχει το τηλέφωνο του δάσκαλου και της σχολής, μαζί με την φυσική τοποθεσία
- (Optional) διακόσμηση της HomePage για τους Members, μία επιλογή είναι να μπεί το calendarView σε μία άλλη Activity εκτός του MemberFragment.