CS-219 - Computer Engineering Workshop

Complex Engineering Problem (CEP) - Report

Fall Semester 2020

Studestant

Student Assisstant

Built with Firebase

Group Members:

Sohaib Ahmed Abbasi (CS-19096)
Rohan Ahmed (CS-19091)
Abdul Moiz Ali (CS-19087)
Zain Ammad (CS-19079)

Table Of Contents

Project Title

Studestant

Abstract

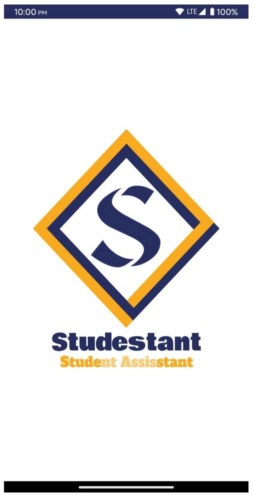
Studestant, short for Student-Assistant, is an android app aimed to facilitate students, boost their productivity, and improve their focus, by providing various useful features in one place. It has the features of To-do List, Pomodoro Timer, Formulas Cheat-Sheet, Wikipedia Search, Google search, and a collection of useful links. The app secures your data with its user account system which uses Firebase authentication, and it includes options to sign in with Email ID and Google ID. Moreover, it uses Firebase's Cloud Firestore as its cloud storage system.

Details

Launcher Icon and Splash Screen

This is how our app's icon looks like in a typical launcher (see image on right).

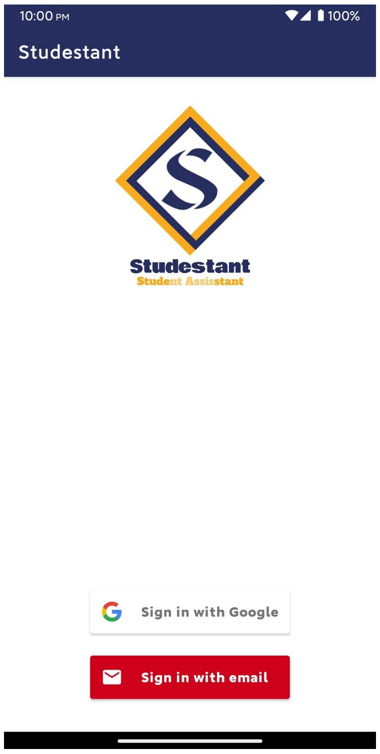
Tapping on the icon opens up the app where you're first greeted with the following splash screen (see image below):

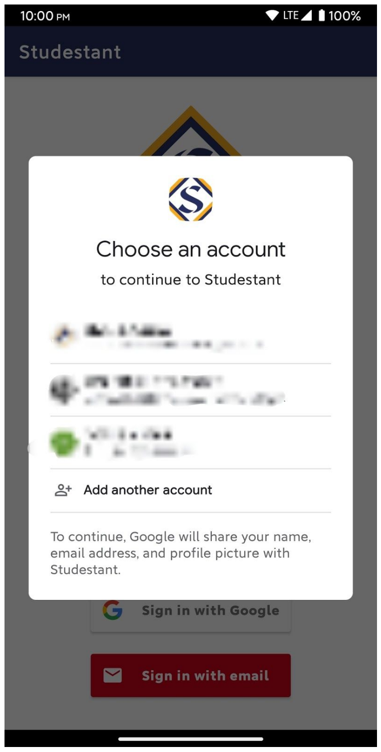


Login Screen

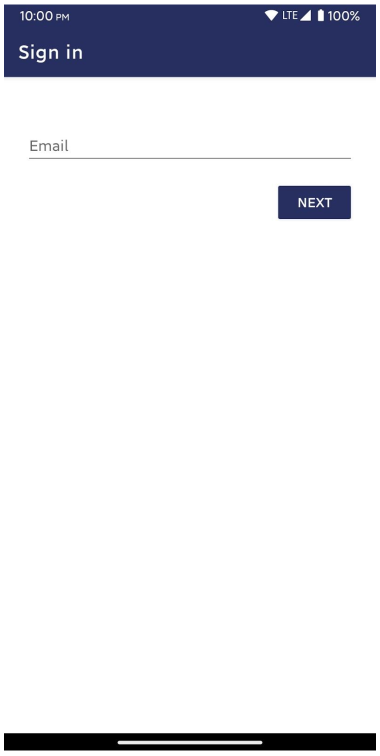
The app uses Firebase for user authentication as well as the login UI. It has the option to sign in with Google ID and email ID. Tapping the Sign in with Google button opens up a menu with a list of accounts already signed in on the device as well as an option to add another account. Tapping the Sign in with email button opens up an input field where you can type in your email

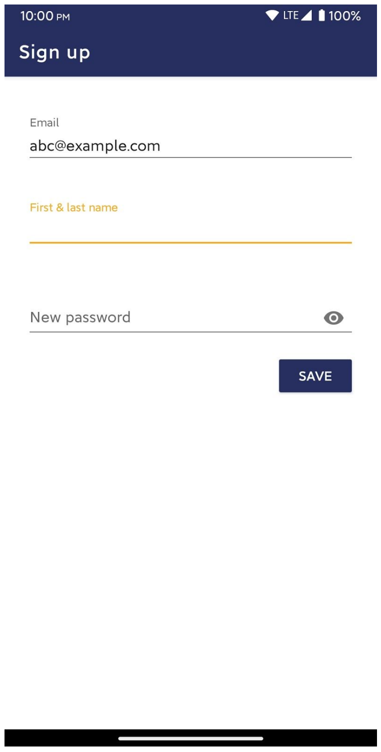
ID. If you type in an email ID that isn't registered with the app, you will be prompted for your name and a new password, if you type in an email ID that is registered with the app you'll be prompted for your password; and if you type in an email ID that corresponds to a Google ID that has been used to log in to the app before, you will be asked to sign in with that google ID.





To continue, Google will share your name, email address, and profile picture with Studestant.

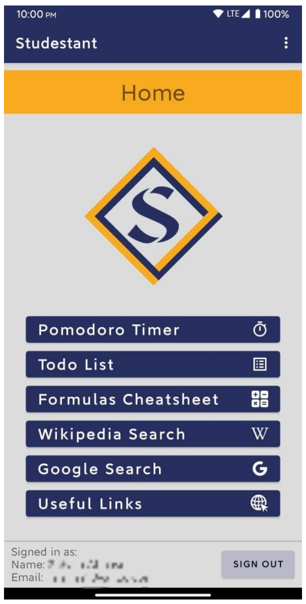




Home Screen

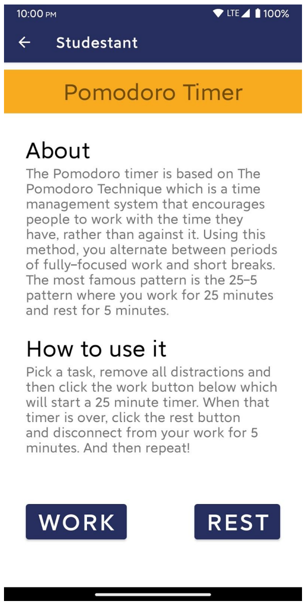
After logging in, the following home screen opens up (see image on right):

The home screen consists of an options menu in the top right corner which gives you the option to sign out; the app logo; buttons leading to the different features of the app; name and email ID of the signed-in user in the bottom right and a sign out button in the bottom left.



Pomodoro Timer

This feature helps you to follow the Pomodoro timer which follows the Pomodoro technique which is a technique to improve focus and productivity by having alternate periods of work and rest. It follows the 25-5 pattern of the Pomodoro technique (25 minutes work, 5 minutes rest). First, a screen with details of the Pomodoro timer and how to use it alongside buttons to start work and rest timer is displayed. Tapping the work button starts a 25-minute timer whereas tapping the rest button starts a 5-minute timer; both of these timers end with an alarm sound.

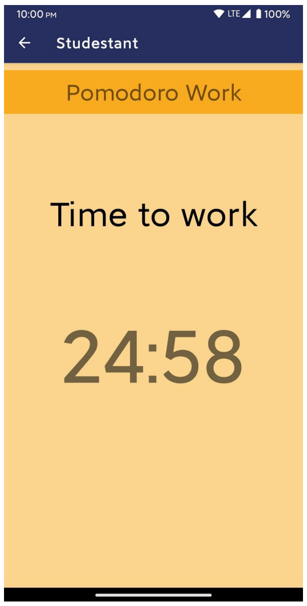


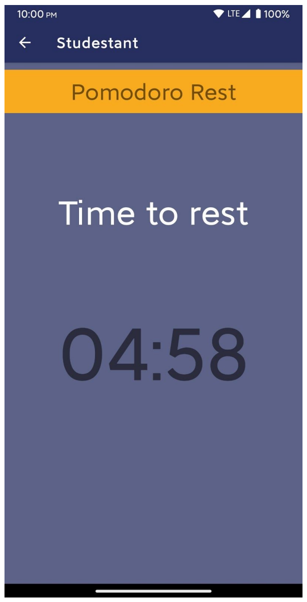
About

The Pomodoro timer is based on The Pomodoro Technique which is a time management system that encourages people to work with the time they have, rather than against it. Using this method, you alternate between periods of fully-focused work and short breaks. The most famous pattern is the 25–5 pattern where you work for 25 minutes and rest for 5 minutes.

How to use it

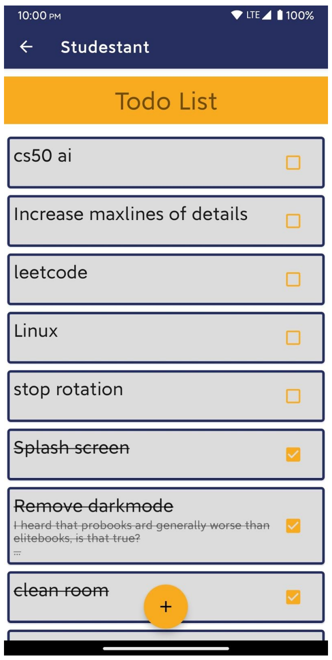
Pick a task, remove all distractions and then click the work button below which will start a 25 minute timer. When that timer is over, click the rest button and disconnect from your work for 5 minutes. And then repeat!





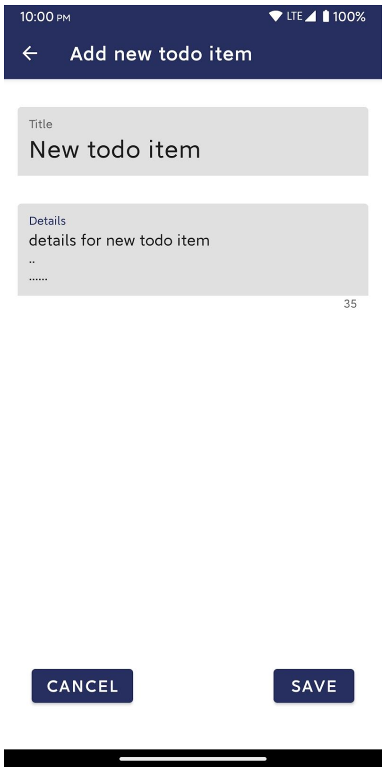
Todo List

This is to be used to store a list of upcoming tasks that one has to do with optional details of those tasks. It uses the Firestore database to store the information of every user's to-do items. Tapping the "+" button at the bottom of the screen gives you the option to make a new to-do item with a title and optional details. Tapping on any to-do item opens up a screen where you can see the full to-do item, edit it and delete it. A to-do item can be marked as done or not done by tapping the checkbox on the right of the respective item; marking as done strikethroughs the title and details text.



Todo List

- cs50 ai ☐
- Increase maxlines of details ☐
- leetcode ☐
- Linux ☐
- stop rotation ☐
- ~~Splash screen~~ ✓
- ~~Remove darkmode~~ ✓
  ~~I heard that probooks ard generally worse than elitebooks, is that true?~~
- ~~clean room~~ ✓

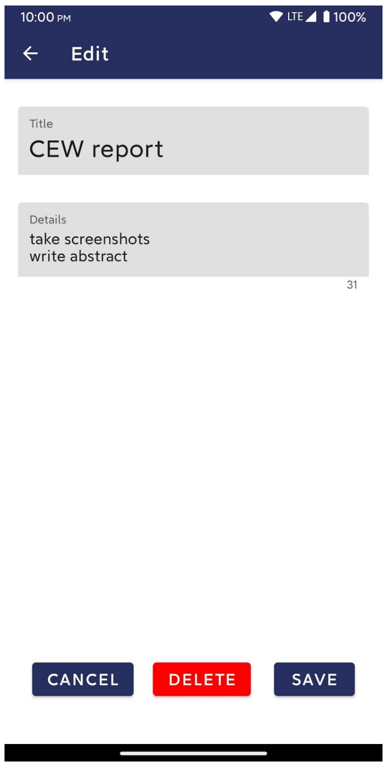


Add new todo item

Title
New todo item

Details
details for new todo item
..
......

35

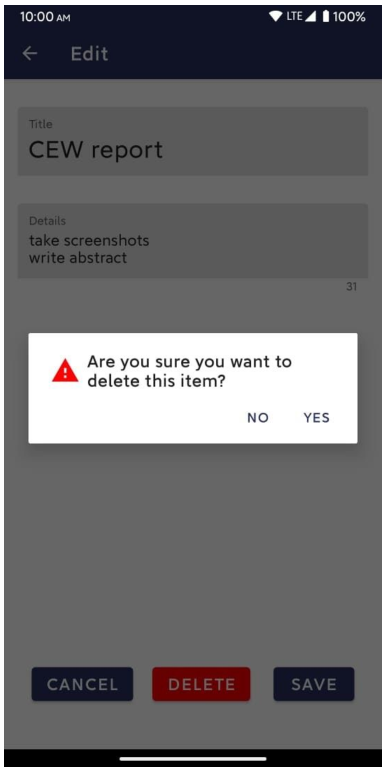


Edit

Title
CEW report

Details
take screenshots
write abstract

31



Edit

Title
CEW report

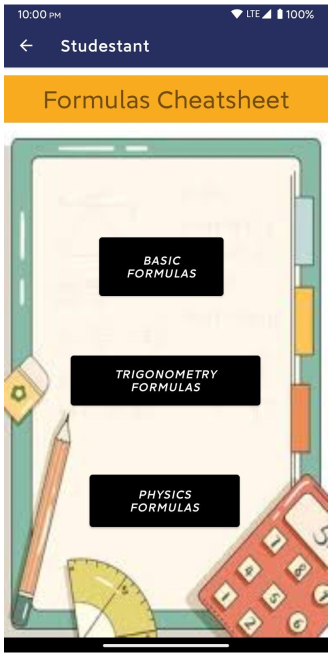
Details
take screenshots
write abstract

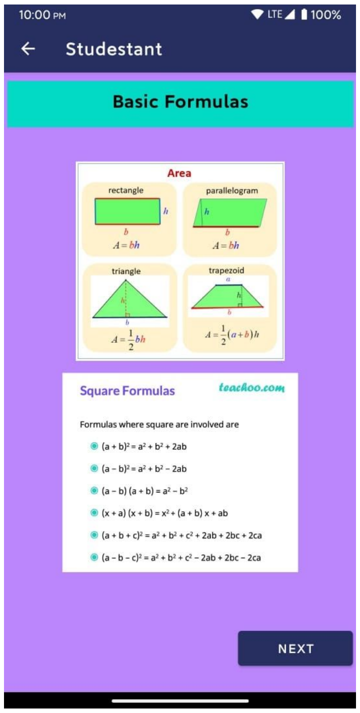
31

Are you sure you want to delete this item?

Formulas Cheatsheet

This part of the app features a bunch of formulas categorized under basic formulas, trigonometric formulas, and physics formulas that you can quickly lookup. It has a screen with buttons that lead you to screens having the respective formula categories.





Area

rectangle $A = bh$

parallelogram $A = bh$

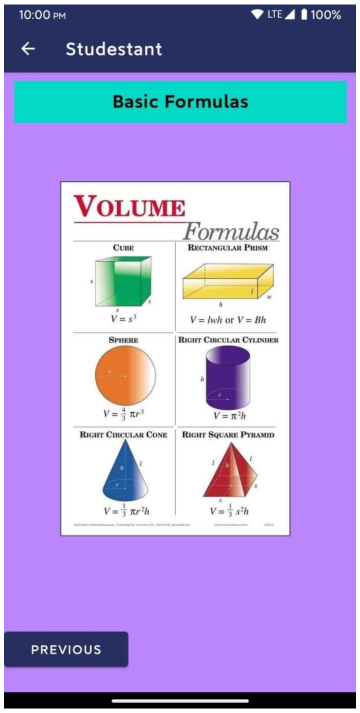
triangle $A = \frac{1}{2}bh$

trapezoid $A = \frac{1}{2}(a+b)h$

Square Formulas teachoo.com

Formulas where square are involved are

- $(a + b)^2 = a^2 + b^2 + 2ab$
- $(a - b)^2 = a^2 + b^2 - 2ab$
- $(a - b)(a + b) = a^2 - b^2$
- $(x + a)(x + b) = x^2 + (a + b)x + ab$
- $(a + b + c)^2 = a^2 + b^2 + c^2 + 2ab + 2bc + 2ca$
- $(a - b - c)^2 = a^2 + b^2 + c^2 - 2ab + 2bc - 2ca$



VOLUME Formulas

CUBE $V = s^3$

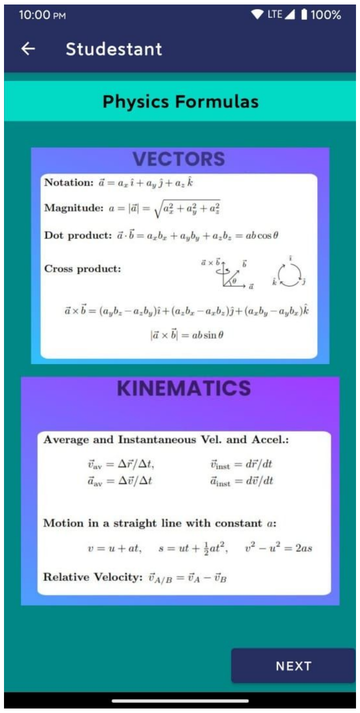
RECTANGULAR PRISM $V = lwh$ or $V = Bh$

SPHERE $V = \frac{4}{3}\pi r^3$

RIGHT CIRCULAR CYLINDER $V = \pi^2 h$

RIGHT CIRCULAR CONE $V = \frac{1}{3}\pi r^2 h$

RIGHT SQUARE PYRAMID $V = \frac{1}{3}s^2 h$



VECTORS

Notation: $\vec{a} = a_x\,\hat{\imath} + a_y\,\hat{\jmath} + a_z\,\hat{k}$

Magnitude: $a = |\vec{a}| = \sqrt{a_x^2 + a_y^2 + a_z^2}$

Dot product: $\vec{a}\cdot\vec{b} = a_x b_x + a_y b_y + a_z b_z = ab\cos\theta$

Cross product:

$$\vec{a}\times\vec{b} = (a_y b_z - a_z b_y)\hat{\imath} + (a_z b_x - a_x b_z)\hat{\jmath} + (a_x b_y - a_y b_x)\hat{k}$$

$$|\vec{a}\times\vec{b}| = ab\sin\theta$$

KINEMATICS

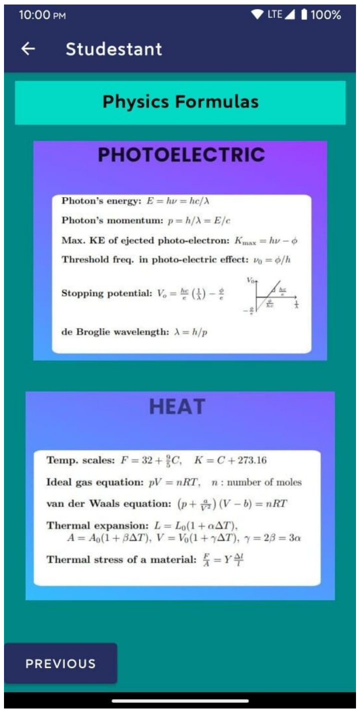
Average and Instantaneous Vel. and Accel.:

$$\vec{v}_{\text{av}} = \Delta\vec{r}/\Delta t, \quad \vec{v}_{\text{inst}} = d\vec{r}/dt$$
$$\vec{a}_{\text{av}} = \Delta\vec{v}/\Delta t \quad \vec{a}_{\text{inst}} = d\vec{v}/dt$$

Motion in a straight line with constant a:

$$v = u + at, \quad s = ut + \tfrac{1}{2}at^2, \quad v^2 - u^2 = 2as$$

Relative Velocity: $\vec{v}_{A/B} = \vec{v}_A - \vec{v}_B$



PHOTOELECTRIC

Photon's energy: $E = h\nu = hc/\lambda$

Photon's momentum: $p = h/\lambda = E/c$

Max. KE of ejected photo-electron: $K_{\max} = h\nu - \phi$

Threshold freq. in photo-electric effect: $\nu_0 = \phi/h$

Stopping potential: $V_o = \frac{hc}{e}\left(\frac{1}{\lambda}\right) - \frac{\phi}{e}$

de Broglie wavelength: $\lambda = h/p$

HEAT

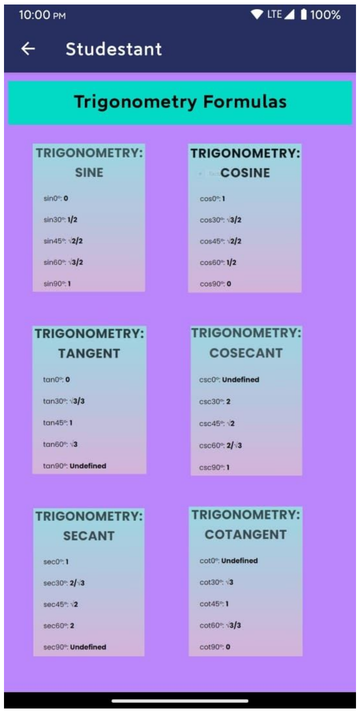
Temp. scales: $F = 32 + \frac{9}{5}C, \quad K = C + 273.16$

Ideal gas equation: $pV = nRT, \quad n$: number of moles

van der Waals equation: $\left(p + \frac{a}{V^2}\right)(V - b) = nRT$

Thermal expansion: $L = L_0(1 + \alpha\Delta T)$, $A = A_0(1 + \beta\Delta T)$, $V = V_0(1 + \gamma\Delta T)$, $\gamma = 2\beta = 3\alpha$

Thermal stress of a material: $\frac{F}{A} = Y\frac{\Delta l}{l}$



TRIGONOMETRY: SINE

sin0°: **0**
sin30°: **1/2**
sin45°: **√2/2**
sin60°: **√3/2**
sin90°: **1**

TRIGONOMETRY: COSINE

cos0°: **1**
cos30°: **√3/2**
cos45°: **√2/2**
cos60°: **1/2**
cos90°: **0**

TRIGONOMETRY: TANGENT

tan0°: **0**
tan30°: **√3/3**
tan45°: **1**
tan60°: **√3**
tan90°: **Undefined**

TRIGONOMETRY: COSECANT

csc0°: **Undefined**
csc30°: **2**
csc45°: **√2**
csc60°: **2/√3**
csc90°: **1**

TRIGONOMETRY: SECANT

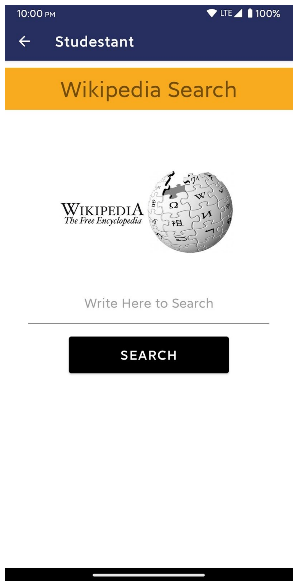
sec0°: **1**
sec30°: **2/√3**
sec45°: **√2**
sec60°: **2**
sec90°: **Undefined**

TRIGONOMETRY: COTANGENT

cot0°: **Undefined**
cot30°: **√3**
cot45°: **1**
cot60°: **√3/3**
cot90°: **0**

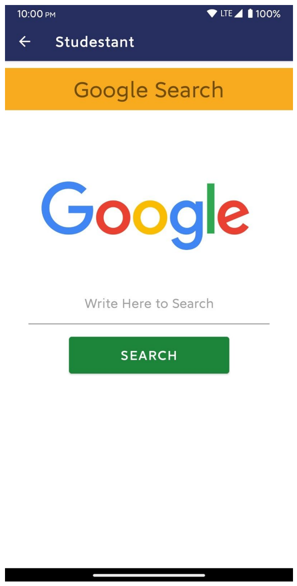
Wikipedia Search

This feature can be used to open up any Wikipedia page you want. It has an input field where you can type the name of the Wikipedia page you want to go to and a button that can be tapped to open up the page you want to go in your mobile's browser. The reason for having this feature in the app is to prevent the user from going into their browser and searching for the Wikipedia page they want because that can easily lead to the user falling to the temptation of going to some other time-wasting / non-productive website or application.



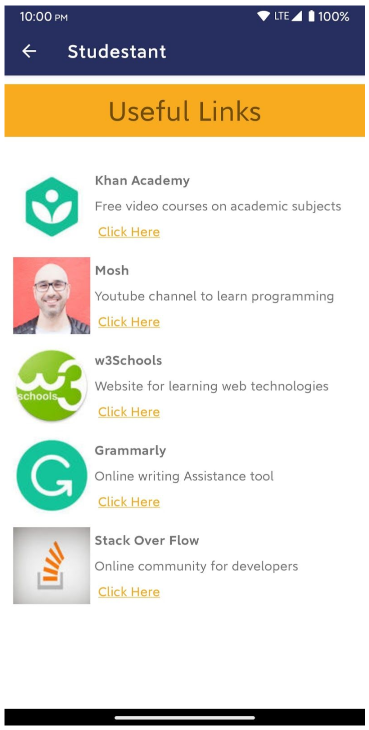
Google Search

Very similar to the Wikipedia search feature, this feature has a similar purpose and functionality. It can be used to type in and open up the google search page of any thing you want to look up. It can particularly be useful to search up specific definitions, formulas, etc.



Useful Links

The useful links section of the app contains a list of links that students may not know about but can be quite useful. Each link can be opened up by clicking its respective “Click here” text.



Khan Academy
Free video courses on academic subjects
Click Here

Mosh
Youtube channel to learn programming
Click Here

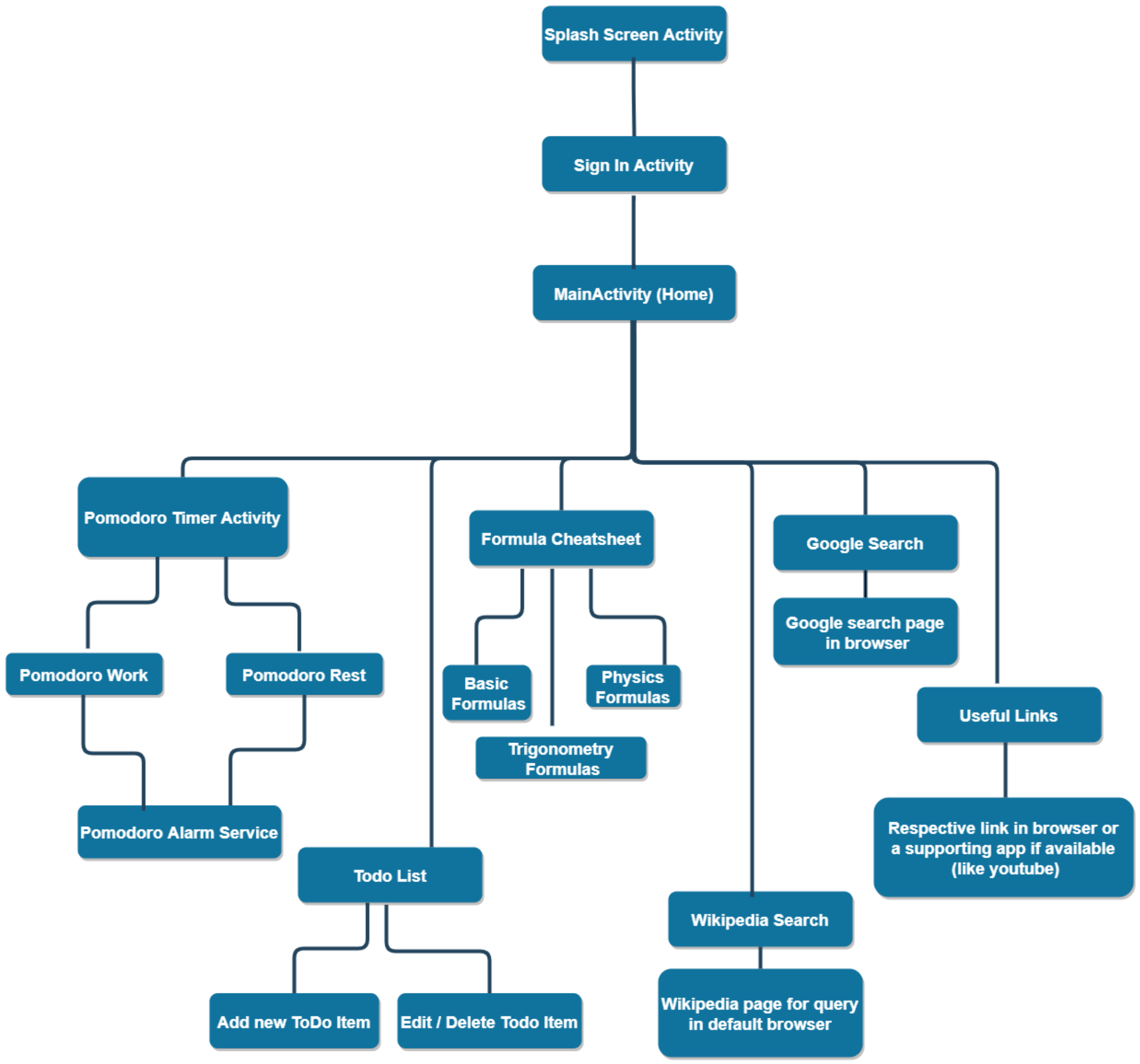
w3Schools
Website for learning web technologies
Click Here

Grammarly
Online writing Assistance tool
Click Here

Stack Over Flow
Online community for developers
Click Here

(Program flow on next page)

Program Flow



Codes of some important classes

MainActivity.java

```
public class MainActivity extends AppCompatActivity {

    private static final int RC_SIGN_IN = 1;
    Button btn_todo_list;
    Button btn_wikipedia_search;
    Button btn_google_search;
    Button btn_formulas_cheatsheet;
    Button btn_pomodoro_timer;
    Button btn_useful_links;
    TextView txtview_userinfo;
    Button btn_signout;

    FirebaseAuth mAuth;  // firebase authentication instance
    FirebaseAuth.AuthStateListener mAuthStateListener;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        startActivity(new Intent(this, SplashScreenActivity.class));

AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO);
// Disable darkmode
        btn_todo_list = findViewById(R.id.btn_todo_list);
        btn_wikipedia_search = findViewById(R.id.btn_wikipedia_search);
        btn_google_search = findViewById(R.id.btn_google_search);
        btn_formulas_cheatsheet =
findViewById(R.id.btn_formulas_cheatsheet);
        btn_pomodoro_timer = findViewById(R.id.btn_pomodoro_timer);
        btn_useful_links = findViewById(R.id.btn_useful_links);
        txtview_userinfo = findViewById(R.id.txtview_userinfo);
        btn_signout = findViewById(R.id.btn_signout);

        mAuth = FirebaseAuth.getInstance();

        btn_todo_list.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this,
TodoListActivity.class);
                startActivity(intent);
            }
        });

        btn_wikipedia_search.setOnClickListener(new View.OnClickListener()
{
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this,
WikipediaSearchActivity.class);
                startActivity(intent);
            }
        });

        btn_google_search.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this,
GoogleSearchActivity.class);
                startActivity(intent);
            }
        });
        btn_formulas_cheatsheet.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this,
FormulasCheatsheetActivity.class);
                startActivity(intent);
            }
        });

        btn_pomodoro_timer.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this,
PomodoroTimerActivity.class);
                startActivity(intent);
            }
        });

        btn_useful_links.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this,
UsefulLinks.class);
                startActivity(intent);
            }
        });

        btn_signout.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                mAuth.signOut();
//                  Auth.GoogleSignInApi.signOut(apiClient);  // check this
for google signout problems
            }
        });

        List<AuthUI.IdpConfig> providers = Arrays.asList(
                new AuthUI.IdpConfig.GoogleBuilder().build(),
                new AuthUI.IdpConfig.EmailBuilder().build());
//                  new AuthUI.IdpConfig.PhoneBuilder().build()

        mAuthStateListener = new FirebaseAuth.AuthStateListener() {
            @Override
            public void onAuthStateChanged(@NonNull FirebaseAuth
firebaseAuth) {
                FirebaseUser current_user = mAuth.getCurrentUser();
                if (current_user == null)  // if user isn't logged in
                {
                    // Create and launch sign-in activity intent
                    startActivityForResult(
                            AuthUI.getInstance()
                                    .createSignInIntentBuilder()
                                    .setAvailableProviders(providers)
                                    .setTheme(R.style.Theme_Studestant)
                                    .setLogo(R.drawable.login)
                                    .setIsSmartLockEnabled(false)
                                    .build(),
                            RC_SIGN_IN);
                }
                updateUI();
            }
        };
    }

    @Override
    protected void onResume() {
        super.onResume();
        // adding authStateListener
        mAuth.addAuthStateListener(mAuthStateListener);
    }

    @Override
    protected void onPause() {
        super.onPause();
        // removing authstateListener
        mAuth.removeAuthStateListener(mAuthStateListener);
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode,
@Nullable Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        if (requestCode == RC_SIGN_IN)  // if result is from firebase
signin activity
        {
            IdpResponse response = IdpResponse.fromResultIntent(data);

            if (resultCode == RESULT_OK) {
                Toast.makeText(this, "Successfully signed in",
Toast.LENGTH_SHORT).show();
            }
            else
            {
                if (response == null)  // user pressed back on login
screen
                {
                    finish(); // close the app
                }
            }
        }
    }

    @Override
    public void onBackPressed() {
        super.onBackPressed();
        finish();
    }


    private void updateUI() {
        FirebaseUser currentUser = mAuth.getCurrentUser();
        if (currentUser == null)  // if not logged in
        {
            txtview_userinfo.setText("User info:\nNot signed in");
            btn_signout.setVisibility(View.GONE);
        }
        else  // if logged in
        {
            btn_signout.setVisibility(View.VISIBLE);
            String name = currentUser.getDisplayName();
            if (TextUtils.isEmpty(name)){
                name = "Not available";
            }
            String email = currentUser.getEmail();
            if (TextUtils.isEmpty(email)){
                email = "Not available";
            }
            // set user name and email in text view
            txtview_userinfo.setText(String.format("Signed in as:\nName:
%s\nEmail: %s", name, email));
        }
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.home_activity_menu, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(@NonNull MenuItem item) {
        switch (item.getItemId())
        {
            case R.id.signout_menu_option:
                mAuth.signOut();
                return true;
            default:
                return super.onOptionsItemSelected(item);
        }

    }
}
```

PomodoroRest.java

```
public class PomodoroRest extends AppCompatActivity {

    public static final long start_time_ms= 300000;
    TextView tv_time_to_rest;
    TextView tv_j_rest;
    CountDownTimer countdown_timer;

    long left_time= start_time_ms;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_pomodoro_rest);

AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO);
// Disable darkmode

        tv_time_to_rest = findViewById(R.id.tv_ttr);
        tv_j_rest= (TextView)findViewById(R.id.tv_pr);

        tv_time_to_rest.setText("Time to rest");

        start_timer();
    }



    public void start_timer() {
        countdown_timer= new CountDownTimer(left_time,1000) {  //1000ms= 1
second interval
            @Override
            public void onTick(long millisUntilFinished) {
                left_time= millisUntilFinished;
                update_countdown_text(); //updated the text view of timer

            }

            @Override
            public void onFinish() {
                Intent intent_notify = new Intent(PomodoroRest.this,
Tone_Service.class);
                startService(intent_notify);  //play the notification
Sound
                tv_time_to_rest.setText("Time's Up!\nGo back and start the
work timer.");
                update_countdown_text();
                new Handler(Looper.myLooper()).postDelayed(new Runnable()
{
                    @Override
                    public void run() {
                        stopService(intent_notify);
                    }
                }, 2500);  // stop music player service after 2.5 seconds
            }
        }.start();

    }
    public void update_countdown_text() {
        int minutes = (int) (left_time / 1000) / 60;
        int seconds = (int) (left_time / 1000) % 60;

        String time_left_format = String.format(Locale.getDefault(),
"%02d:%02d", minutes, seconds);

        tv_j_rest.setText(time_left_format);

    }
}
```

TodoListActivity.java

```
public class TodoListActivity extends AppCompatActivity {
    private CollectionReference todo_collection_reference;

    private TodoAdapter adapter;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_todo_list);

AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO);
// Disable darkmode

        findViewById(R.id.btn_add_note).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(TodoListActivity.this,
TodoAdderActivity.class));
            }
        });

        FirebaseUser current_user =
FirebaseAuth.getInstance().getCurrentUser();

        if (current_user == null)  // if no user logged in
        {
            finish();  // end the activity
        }

        todo_collection_reference = FirebaseFirestore.getInstance().  //
get an instance of firestore database
                collection("users").  // go to the users collection
                document(current_user.getUid()).  // go inside the
document corresponding to current user
                collection("todoItems");  // go to the todoItems
collection inside that user's document

        setUpRecyclerView();
    }
    @Override
    protected void onResume() {
        super.onResume();
        if (FirebaseAuth.getInstance().getCurrentUser() == null)  // if
user not logged in
        {
            finish();  // end activity
        }
        adapter.startListening();
    }

    @Override
    protected void onPause() {
        super.onPause();
        adapter.stopListening();
    }


    private void setUpRecyclerView()
    {
        Query query = todo_collection_reference.orderBy("isDone",
Query.Direction.ASCENDING);

        FirestoreRecyclerOptions<TodoItem> options = new
FirestoreRecyclerOptions.Builder<TodoItem>()
            .setQuery(query, TodoItem.class)
            .build();

        adapter = new TodoAdapter(options);

        RecyclerView recyclerView = findViewById(R.id.recycler_view_todo);
        recyclerView.setHasFixedSize(true);
        recyclerView.setLayoutManager(new LinearLayoutManager(this));
        recyclerView.setAdapter(adapter);

    }

}
```

TodoEditorActivity.java

```
public class TodoEditorActivity extends AppCompatActivity {
    FirebaseUser current_user;
    DocumentReference todo_item_reference;

    AlertDialog.Builder alert_dialog_builder;
    AlertDialog delete_alert_dialog;

    EditText et_todo_editor_title;
    EditText et_todo_editor_details;

    String doc_id;
    String todo_title;
    String todo_details;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_todo_editor);
        setTitle("Edit");

AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO);
// Disable darkmode

        current_user = FirebaseAuth.getInstance().getCurrentUser();
        if (current_user == null)  // if not logged in
        {
            finish();
        }

        Intent intent = getIntent();

        // validation
        if ( ! (intent.hasExtra("todo_document_id") &&
                intent.hasExtra("todo_title") &&
intent.hasExtra("todo_details")))
        {
            Toast.makeText(this, "Error, couldn't find the todo item.",
Toast.LENGTH_SHORT).show();
            finish();
        }

        doc_id = intent.getStringExtra("todo_document_id");
        todo_title = intent.getStringExtra("todo_title");
        todo_details = intent.getStringExtra("todo_details");

        todo_item_reference =
                FirebaseFirestore.getInstance().  // get an instance of
firestore database
                        collection("users").  // go to the users
collection
                        document(current_user.getUid()).  // go inside the
document corresponding to current user
                        collection("todoItems").  // go to the todoItems
collection inside that user's to-dolist collection
                        document(doc_id);  // go to the specific to-do
item document

        et_todo_editor_title =
findViewById(R.id.edittxt_todo_editor_title);
        et_todo_editor_details =
findViewById(R.id.edittxt_todo_editor_details);

        et_todo_editor_title.setText(todo_title);
        et_todo_editor_details.setText(todo_details);

        // building alert dialog for delete
        alert_dialog_builder = new AlertDialog.Builder(this);
        alert_dialog_builder.setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                todo_item_reference.delete();
                Toast.makeText(TodoEditorActivity.this, "Item deleted",
Toast.LENGTH_SHORT).show();
                finish();
            }
        });
        alert_dialog_builder.setNegativeButton("No", new
DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                // pass
            }
        });
        alert_dialog_builder.setTitle("Are you sure you want to delete
this item?");
        alert_dialog_builder.setIcon(R.drawable.ic_warning);
        delete_alert_dialog = alert_dialog_builder.create();

        findViewById(R.id.btn_todo_editor_cancel).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });

        findViewById(R.id.btn_todo_editor_save).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                saveChanges();
            }
        });

        findViewById(R.id.btn_todo_editor_delete).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                delete_alert_dialog.show();
            }
        });
    }


    private void saveChanges() {
        if (current_user == null)  // if no user logged in
        {
            finish();  // end the activity
        }
        String updated_todo_title =
et_todo_editor_title.getText().toString().trim();
        if (updated_todo_title.equals(""))
        {
            Toast.makeText(this, "Can't save a todo item without a
title.", Toast.LENGTH_SHORT).show();
            return;
        }

        String updated_todo_details =
et_todo_editor_details.getText().toString().trim();

        // update the to-do item in cloud firestore (which will also
automatically update the recyclerview)
        todo_item_reference.update("title", updated_todo_title, "details",
updated_todo_details);

        Toast.makeText(this, "Successfully saved changes",
Toast.LENGTH_SHORT).show();
        finish();
    }


}
```

FormulasCheatSheetActivity.java

```
public class FormulasCheatsheetActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_formulas_cheatsheet_activty);

AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO);
// Disable darkmode
    }

    public void basic_formulas(View view) {
        Intent intent=new
Intent(FormulasCheatsheetActivity.this,Basic_Formulas.class);
        startActivity(intent);
    }

    public void trigonometry_formulas(View view) {
        Intent intent=new
Intent(FormulasCheatsheetActivity.this,Trigonometry_Formulas.class);
        startActivity(intent);
    }

    public void physics_formulas(View view) {
        Intent intent=new
Intent(FormulasCheatsheetActivity.this,Physics_Formulas.class);
        startActivity(intent);
    }
}
```

WikipediaSearchActivity.java

```
public class WikipediaSearchActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_wikipedia_search);

AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO);
// Disable darkmode
    }


    public void btnclick(View view) {

        EditText et_wiki= (EditText)findViewById(R.id.editText_wiki);
        String et_w=et_wiki.getText().toString();

        if (et_w.isEmpty()) {
            Toast.makeText(this,"You did not
write",Toast.LENGTH_SHORT).show();

        }
        else {

            String url = "https://en.wikipedia.org/wiki/" + et_w; //making
url which is going to be redirected to browser


            Intent intent = new Intent(Intent.ACTION_VIEW,
Uri.parse(url));
            startActivity(intent);
        }
    }
}
```

UsefulLinks.java

```
public class UsefulLinks extends AppCompatActivity {


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_useful_links);

AppCompatDelegate.setDefaultNightMode(AppCompatDelegate.MODE_NIGHT_NO);
// Disable darkmode

        TextView tv_moshLink = findViewById(R.id.tv_moshLink);
        tv_moshLink.setMovementMethod(LinkMovementMethod.getInstance());

        TextView tv_w3Link = findViewById(R.id.tv_w3Link);
        tv_w3Link.setMovementMethod(LinkMovementMethod.getInstance());

        TextView tv_stackLink = findViewById(R.id.tv_stackLink);
        tv_stackLink.setMovementMethod(LinkMovementMethod.getInstance());

        TextView tv_grammarlyLink = findViewById(R.id.tv_grammarlyLink);

tv_grammarlyLink.setMovementMethod(LinkMovementMethod.getInstance());

        TextView tv_kAcadLink = findViewById(R.id.tv_kAcadLink);
        tv_kAcadLink.setMovementMethod(LinkMovementMethod.getInstance());

    }
}
```